THE ROLE OF HUMAN MOTION IN HUMAN-ROBOT INTERACTION

By

ANAND RAJIV THOBBI

Bachelor of Engineering in Electronics and
Telecommunications
University of Pune
Pune, Maharashtra, India
2009

Submitted to the Faculty of the
Graduate College of
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
December, 2011

THE ROLE OF HUMAN MOTION IN HUMAN-ROBOT INTERACTION

Thesis Approved:

Weihua Sheng, Ph.D.

Thesis Advisor

Guoliang Fan, Ph.D.

Qi Cheng, Ph.D.

Sheryl A. Tucker, Ph.D.

Dean of the Graduate College

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

According to Thrun [1], the broad field of robotics can be classified into three categories as per their application domains - industrial robotics, professional service robotics, and personal service robotics. Industrial robotics refer to the robots deployed in strictly structured industrial settings. Such robots have limited reprogrammability and little or no interaction with humans. Professional service robots are employed in special environments such as hospitals and often have very specific tasks, such as performing surgeries. These robots also have limited interaction with humans. Robotics in these two domains is well developed and has been applied commercially with great success.

Inspired by the success of these two fields, there is a steeply growing interest in the robotics research community towards developing general purpose personal service robots which can reside with and assist humans in daily tasks. Such robots are expected to adapt to any environment, communicate well with humans, perform tasks autonomously as well as collaboratively with humans. Apart from classical challenges in robotics such as perception, localization, navigation and manipulation, a new challenge involved in deploying personal service robots is that of *human-robot interaction* (HRI) [2].

1.1 Human-Robot Interaction (HRI)

The central theme in a variety of science fiction works, including Capek's play *Rossums Universal Robots*, is how robots would live, interact and communicate with humans.

Human-Robot Interaction (HRI) is a sub-field of robotics that tries to understand and shape the way humans and robot would interact with each other.

According to a survey on HRI by Goodrich *et al.* [3], the major attributes that affect HRI include

- Nature of information exchange which are specified by various communication modalities
- Level of autonomy which decides whether the robot is completely autonomous, partially teleoperated or fully teleoperated
- Structure of the team which decides the role of the agents in the interaction
- Adaptation, learning, and training of both people and robots
- Shaping tasks to make them simpler for both humans and robots

It is easy to imagine that information exchange or human-robot communication (HRC) plays a major role in determining the efficacy of the interaction.

1.2 Human-Robot Communication

Humans can communicate with each other, very effectively using language, sound, hand gestures etc. Apart from such explicit modes of communication, humans can also communicate implicitly, by employing body language, facial expressions or by even performing actions (*actions speak louder than words*).

Tremendous amount of progress has been made in developing natural language capabilities for robots. Progress has also been made in the vision community which allows robots to sense hand gestures and facial expressions. In addition, robots can also communicate using a variety of devices such as keyboards, visual displays, haptic interfaces etc. Apart from such communication modalities, in order to make HRI truly effective and convenient, we also need to enable robots to understand the implicit

mode of communication. Let us take for example a robotic *butler* who can assist us with our daily lives. The butler would be able to perform a task if he is told to do so explicitly using speech or hand gestures. However, the butler would be really effective if he could infer our mental or physical state based on various features such as facial expressions, our bodily motions or even the current context, and proactively provide assistance. To make implicit communications possible, such robots need to acquire situational knowledge, which gives them the ability to infer/predict human intention based on their observations.

1.3 Thesis Overview

The aim of this work is to investigate how human motion can be used for communicating with robots, both explicitly and implicitly. The term *human motion* is very general since humans can use their motion for communication in various ways. In this work we focus on -

- Arm gestures - observing the human arm's joint angles for learning to imitate human arm gestures
- Demonstrating tasks - observing the movement of human's hands in the workspace for learning to perform tasks by demonstration
- Performing collaborative tasks - inferring human intent by observing his motion while working with him.

Figure 1.1 illustrates the various segments of this work. In the first two scenarios, the robot is explicitly shown the gestures and the tasks, which is to be used for learning them. In the third scenario, the robot has to watch human's motion while working with him, and infer his intent.

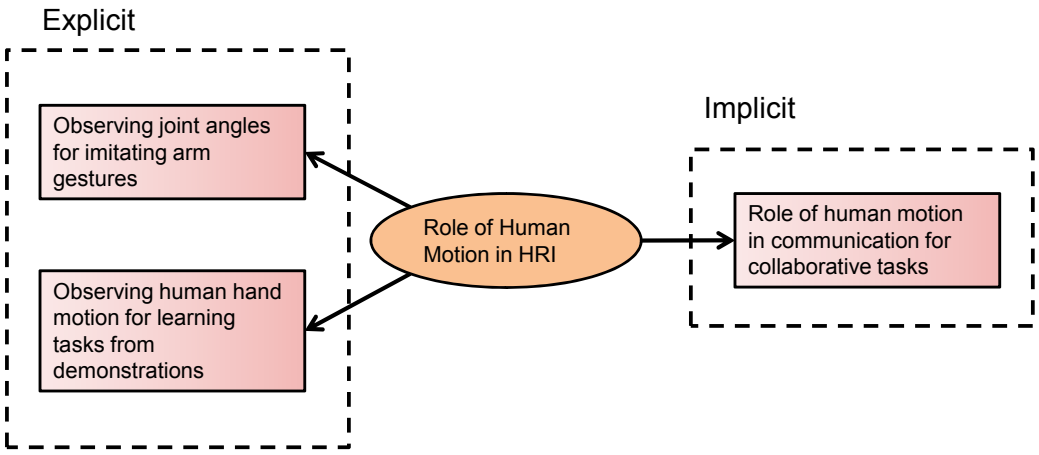


Figure 1.1: The role of human motion in HRI

1.3.1 Explicit Communication By Motion

Learning from demonstrations

Learning from Demonstrations (LfD) is a very good example of where explicit communication is needed between a human and a robot [4]. Traditionally, robots have to be programmed systematically to carry out specific tasks. But if the robot has to work in a home scenario with non-expert users, it is essential for the to learn a variety of tasks from humans without having the need of humans to program them using sophisticated programming techniques. The learning by imitation paradigm offers a novel method to teach robots various skills [5]. This principle has taken inspiration from nature, since learning by imitation is one of the most fundamental qualities possessed by living beings in general and human beings in particular. According to studies in neuroscience [6] and evolution theory, it is this ability which has helped human civilization advance with such rapidity.

Imitating arm gestures

Humans often use hand gestures or arm gestures for communication. Arm gestures could be as simple as waving arms for signalling or as complex as the gestures used

by referees in sports like football. It would be very useful if robots could learn to recognize as well as perform arm gestures, given a few demonstrations. If robots can characterize the useful information from the demonstrated arm gestures, both problems could be solved. In this work we consider a scenario where the robot observes the joint angles of the human arm, while he is performing a specific gesture. From multiple demonstrations of the same gesture, the robot tries to create a generalized trajectory for that specific gesture. This trajectory could be used by the robot to replay the gesture. Chapter 2 details the framework that would allow robots to imitate arm gestures. We also consider the problem of how to tackle missing data segments, if present in the training data.

Learning to perform tasks from demonstrations

Apart from teaching robots to imitate humans, they can also be taught to perform various visuo-motor tasks by human demonstration. The problem is more challenging, since the state of the environment need not be exactly the same as when the task was demonstrated. The structure of the human and the robot can also be quite different. Hence, simply imitating the demonstrator would not accomplish the objective of performing the task.

For learning to perform tasks by demonstrations, humans tend to observe the movement of the demonstrator's hands with respect to the objects of interest in the workspace, as opposed to observing joint angles of the demonstrator [7]. In this work, we consider a scenario of a humanoid robot learning to perform a visuo-motor task, such as grasping a table at a specific point, by observing motion of the human hand with respect to the table. Again, multiple demonstrations of the same task are needed, from which critical information is extracted. Chapter 3 details the application of the GMM/GMR encoding technique [8] to solve this problem.

1.3.2 Implicit Communication by Motion

According to a survey by Reed *et al.* [9], humans have been found to convey their intent by using their motion or actions in a collaborative task. For example, consider a scenario where two humans are transporting a table from one place to another. When two humans are working on such a collaborative task, their roles as being a leader or a follower are not clearly marked. Suppose one member of the team wants to move the table to the left, he will go ahead and move his end of the table towards the left, anticipating that his team-mate would oblige. If the other team-mate feels that he move helps the team to reach their final objective, then he would oblige and also start moving to the left. This is the kind of communication that not only makes performing collaborative tasks possible, but also effortless. In this work, we consider a physical HRI task, where a human and a robot have to move a table to a random height and place it back down. Since the robot is not assigned a mere follower role, the task is non trivial. Apart from simply reacting to changes in the pose of the table, the robot should also be able to predict human's motion, infer his intent and take proactive actions to keep the table exactly horizontal throughout the task. For making this possible, we propose a framework that allows the robot to determine its own role in the cooperative task and take actions which are appropriate to its role. Chapter 4 presents the proposed framework.

We also investigate how this framework can be extended to other collaborative task, where the prediction and inference is for a long term. We consider a scenario where a human and a robot would work together to assemble various ingredients to prepare a dish in a cooperative cooking scenario. Chapter 5 discusses the cooperative cooking scenario. Finally, chapter 6 presents the conclusions derived from this work.

CHAPTER 2

LEARNING TO IMITATE ARM GESTURES

Humans use their motion to communicate explicitly when they signal to others using various hand or arm gestures. During human-robot interaction, it would be very useful if humans and robots could communicate explicitly using arm gestures. Hence it is very desirable that robots learn to recognize and perform arm gestures. The gesture recognition part has been extensively researched upon by both the robotics and the computer vision community. However, there is relatively less work on teaching robots to perform arm gestures. In this work, we investigate how the learning from demonstrations or imitation learning concept can be used for teaching a humanoid robot to perform various arm gestures.

For learning to perform an arm gesture, the robot is told to observe the demonstrations of the human, specifically the joint angle trajectories of his arm. For learning a particular gesture, multiple demonstrations of the same gesture are required so that the robot can extract essential features of the gesture and learn it, instead of simply copying it [10]. The major issues to address are feature extraction, data representation and data generalization.

We focus on developing a platform to implement and evaluate the imitation learning framework. The joint angle data for the arm gesture is collected using a marker based optical tracking system. The human arm is modeled as being made of 2 segments - the upper arm and the lower arm. We obtain the joint angle trajectories of the shoulder and elbow using the pose of the upper arm and lower arm respectively. For our experimentation, we have considered various hand gestures which explore a

variety of human arm joint angles such as *sweeping motion, knocking motion* and *writing alphabets*.

The proposed imitation learning framework aims at obtaining a generalized representation of a particular arm gesture from multiple demonstrations of the same. Dynamic Time Warping [11] is applied to the recorded joint angle trajectories in order to temporally align the trajectories with minimum error. We generalize these aligned trajectories by weighted averaging. We pass the averaged trajectory through a low pass filter to achieve a smooth trajectory, so that the robot's arm movements are free of jerks. The technique has to be slightly modified when the demonstrated trajectories contain missing data segments. Section 2.5 details this case.

2.1 Related Work

Hidden Markov Models have been successfully applied to the problem of encoding the trajectories and recognizing the arm gestures in [10]. Pollard *et al.* [12] have proposed a control systems based approach to learn and reproduce a particular gesture while satisfying the joint and velocity limits. Dariush *et al.* [13] have described motion primitives in a task space and formulated a solution to track the task descriptor while satisfying the constraints of maintaining balance, collision-avoidance, limiting joint angles and velocities on the ASIMO humanoid robot. The use of B-spline wavelets for representing a trajectory has been explored in [14]. For imitation, a novel idea of connecting some control points on the humanoid to the optical markers attached on the human, using virtual springs has been proposed in [15]. In [16], The problem of imitating human motion has been formulated and solved as an optimization problem with the physical limits of the robot as the constraints .

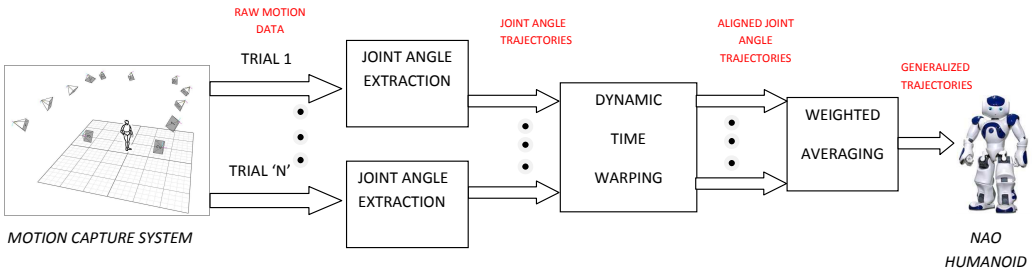


Figure 2.1: Arm gesture imitation system diagram

2.2 Platform for Imitation Learning

Figure 2.1 shows the proposed system for imitation learning. Data is collected from the motion capture system. The joint angle trajectories are extracted from the motion capture data. These trajectories are used as the training data for the imitation learning framework. The proposed framework involves temporally aligning the trajectories using DTW and then deriving their generalized representation. These trajectories are smoothed and applied to the robot. To evaluate the merit of the proposed imitation learning framework, the motion performed by the robot is captured using the motion capture system and compared to the generalized trajectories.

2.2.1 Motion Capture System

The motion capture system used for our experiments is the Vicon MX motion capture system [17]. It is one of the most advanced optical motion capture systems available commercially. The system consists of 12 Vicon T-40 cameras. Each camera can capture a 10 bit grayscale image at a resolution of 4 megapixels. We can capture data at speeds upto 370 frames per seconds. The system is equipped with sophisticated dynamic reconstruction algorithms for real time tracking. The motion capture cameras are connected by a gigabit ethernet port. With the given system, we can track any optical marker within a tolerance of 7 mm. We can create rigid bodies which

are nothing but markers attached to a solid body in a specific pattern. The Vicon Tracker software is used for capturing the rigid-body data. The algorithms used in Tracker are optimized for tracking rigid bodies.

2.2.2 Humanoid Robot

The Nao humanoid robot shown in Figure 2.2, manufactured by Aldebaran Robotics, France, is used for the experiment [18]. The robot has 25 degrees of freedom, realized by a number of motors and actuators. The robot is equipped with a variety of sensors such as 2 cameras, 4 microphones, 2 sonar distance sensors, inertial sensors, tactile sensors and force sensors on the feet. The robot can express itself using a variety of devices such as motion, in built speech synthesizer and a number of LEDs. The battery can last about 1.5 hours. The robot is equipped with an x86 AMD Geode processor running at 500MHz. The processor runs a proprietary embedded linux OS based on the Open Embedded distribution. The robot is equipped with a middle-ware called NaoQi which allows programmers to develop applications for the robot using a variety of languages such as C++ and Python.

Figure 2.2: The Nao humanoid robot

For this experiment, we use only 4 degrees of freedom in the right arm of the robot. The robot has 2 degrees of freedom in the shoulder joint and 2 degrees of freedom in the elbow joint.

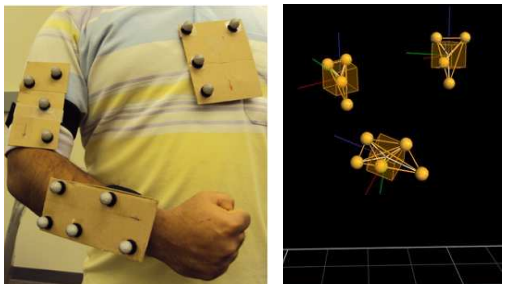
2.3 Imitation Learning Methodology

2.3.1 Human Arm Modelling

We model the human arm, as consisting of two segments, the upper arm and the lower arm. It is clear that the motion of the upper arm and lower arm are governed by the shoulder joint and the elbow joint respectively. Thus, we can estimate the joint angles of the shoulder joint and elbow joint if the orientation of the upper arm and lower arm are known. The following sub-sections present a brief overview of the anatomy of the human arm [19].

Shoulder Joint

The glenohumeral joint or the shoulder joint is a ball and socket joint. It has two significant degrees of freedom. Using this joint, the arm can perform the arm abduction-adduction and arm flexion-extension motions. There can also be the medial and lateral rotations for the shoulder joint, but it is usually very small. In the Nao humanoid robot, there are only two degrees of freedom at the shoulder. Hence the rotation of the shoulder joint has to be ignored. There are also other degrees of freedom in the human arm at the sterno-clavicular joint which gives rise to scapular retraction, protrusion, depression and elevation motions. These movements give rise to expressions such as shrugging or slumping. Unfortunately, humanoid robots cannot be as flexible as the real human arm, and hence these degrees of freedom also have to be ignored.



(a) Markers placed on subject for collecting joint angle data

(b) Rigid bodies (marker plates) as seen by Vicon Tracker

Figure 2.3: Markers for motion capture

Elbow Joint

The elbow joint consists of three independent joints. The humero-ulnar joint is a hinge joint. It allows for the flexion and extension of the lower arm. The proximal radio-ulnar joint is responsible for pronation and supination. The humero-radial joint is an arthroidal joint and does not contribute to the movement of the lower arm.

2.3.2 Data Acquisition

The markers for motion capture are placed on the human subject as shown in Figure 2.3(a). We create a rigid body plate consisting of atleast 4 markers. Three such plates are attached to torso, upper arm and lower arm as shown in Figure 2.3(b). From Tracker, we can get the rotation matrix for each rigid body with respect to the world co-ordinate system.

Let R_T^W, R_U^W, R_L^W be the rotation matrices of the torso plate, upper arm plate and the lower arm plate respectively with respect to the global co-ordinate system. In order to compute the joint angles, we need to estimate the orientation of the upper with respect to the human torso and the orientation of lower arm with respect to the upper arm. Hence, we convert the reference co-ordinate systems as :

$$R_U^T = R_U^W (R_T^W)^{-1}. \tag{2.1}$$

Equation (1) gives us the rotation matrix of the upper arm marker plate with respect to the torso marker plate.

$$R_L^U = R_L^W (R_U^T)^{-1}. \tag{2.2}$$

Equation (2) gives us the rotation matrix of the lower arm marker plate with respect to the upper arm marker plate. Let us represent a rotation matrix R as

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

We can derive the roll, pitch and yaw angles for the marker plates from their rotation matrices. We use the X-Y-Z fixed angle system which is represented by the set of rotations γ, β, α applied to X, Y and Z axes respectively. Given the rotation matrix R, these angles can be derived as,

$$\begin{aligned} \beta &= Atan2(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2}), \\ \alpha &= Atan2(r_{21}/c\beta, r_{11}/c\beta), \\ \gamma &= Atan2(r_{32}/c\beta, r_{33}/c\beta). \end{aligned} \tag{2.3}$$

The solution degenerates when $\beta = \pm 90°$. If $\beta = 90°$, the solution is given by $\beta = 90°, \alpha = 0, \gamma = Atan2(r_{12}, r_{22})$. If $\beta = -90°$, the solution is given by $\beta = -90°, \alpha = 0, \gamma = -Atan2(r_{12}, r_{22})$.

These angles would define the orientation of the respective arm segments they are attached to. The yaw and pitch angles of the upper arm marker plate with respect to

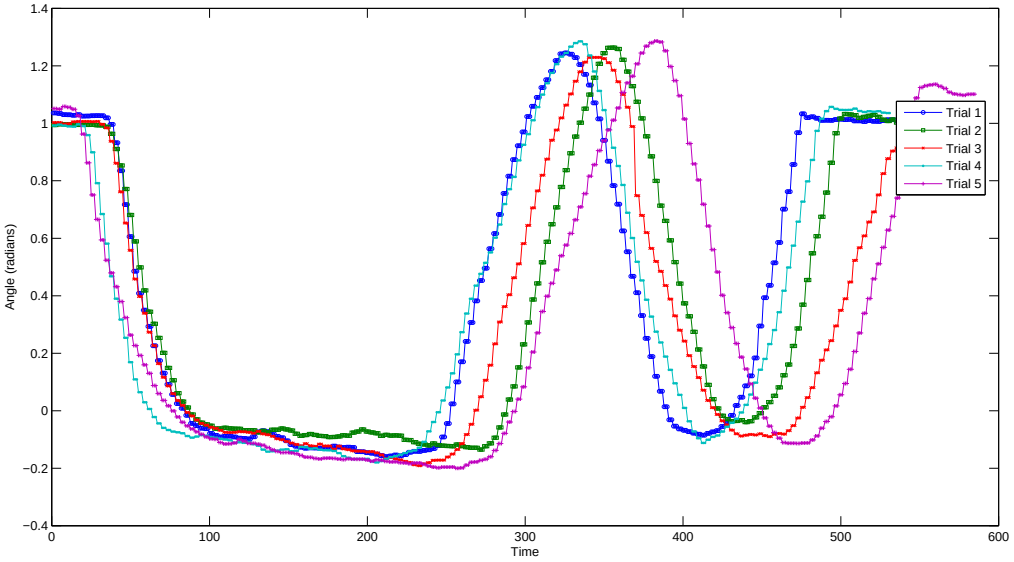


Figure 2.4: Shoulder yaw angle trajectories for sweep motion for 5 demonstrations

torso defines the shoulder yaw and shoulder pitch angles respectively. The yaw and roll angles of the lower arm marker plate with respect to the upper arm marker plate defines the elbow yaw and elbow roll angles respectively. Thus, we extract the joint angles of the human arm.

2.3.3 Data Analysis

Figure 2.4 shows the raw data obtained for the shoulder yaw angle for the sweep gesture performed by the subject five times. Upon visually inspecting the data, the following conclusions about the data can be drawn:

Noise

It can be seen that that the noise induced in the measurements is very small, since the motion capture system used gives extremely accurate data. But in the real world scenario, some noise maybe induced in the measurements which can be removed using a low pass filter.

Variation in Data

It can be seen that every trajectory differs from others in phase. Also, the subject may perform the same gesture with different pace. Hence, we must apply some technique which will bring temporal coherence between the trajectories, while minimizing the difference between them caused by time normalization. This problem is similar to the one, faced in speech recognition systems. Untill more recently, Dynamic Time Warping (DTW) was a standard algorithm used for speech recognition . It has been largely replaced today by Hidden Markov Models(HMM). But, it is clear that speech signals are much more complex than the observed joint angle trajectories. The use of HMMs would be too costly for such a system. Hence, we propose the use of DTW for our system.

2.3.4 Dynamic Time Warping

Dynamic Time Warping is an optimum dynamic programming based time normalization algorithm originally intended for use in speech recognition systems [11]. This Dynamic Programming (DP) based matching algorithm gives a non-linear time normalization effect between two signals. A time warping function is defined as a function which maps the time scale of one signal to another. Consider 2 vectors $A = \{a_1, ..., a_i, ...a_I\}$ *and* $B = \{b_1, ..., b_j, ...b_J\}$. We can define a vector $C = \{c_1, ..., c_k, ...c_K\}$ such that,

$$c(k) = (i(k), j(k)). \tag{2.4}$$

$c(k)$ represents a function which realizes a mapping of the time axis of vector A and time axis of vector B, hence called the warping function. As a measure of distance between the two vectors we can define

$$d(c) = d(i, j) = ||a_i - b_j|| \tag{2.5}$$

The summation of distances on the warping function F is defined as

$$E(F) = \sum_{k=1}^{K} d(c(k)) \tag{2.6}$$

The minimum of (8) with respect to F gives us the optimal warping path.

The important restrictions on this path are:

1. Monotonicity : The path must be monotonous

   $i(k-1) \leq i(k)$ *and* $j(k-1) \leq j(k)$

2. Continuity : The path must be continuous

3. Boundary Conditions

   $i(1) = 1, j(1) = 1$ *and*,

   $i(K) = I, j(K) = J$

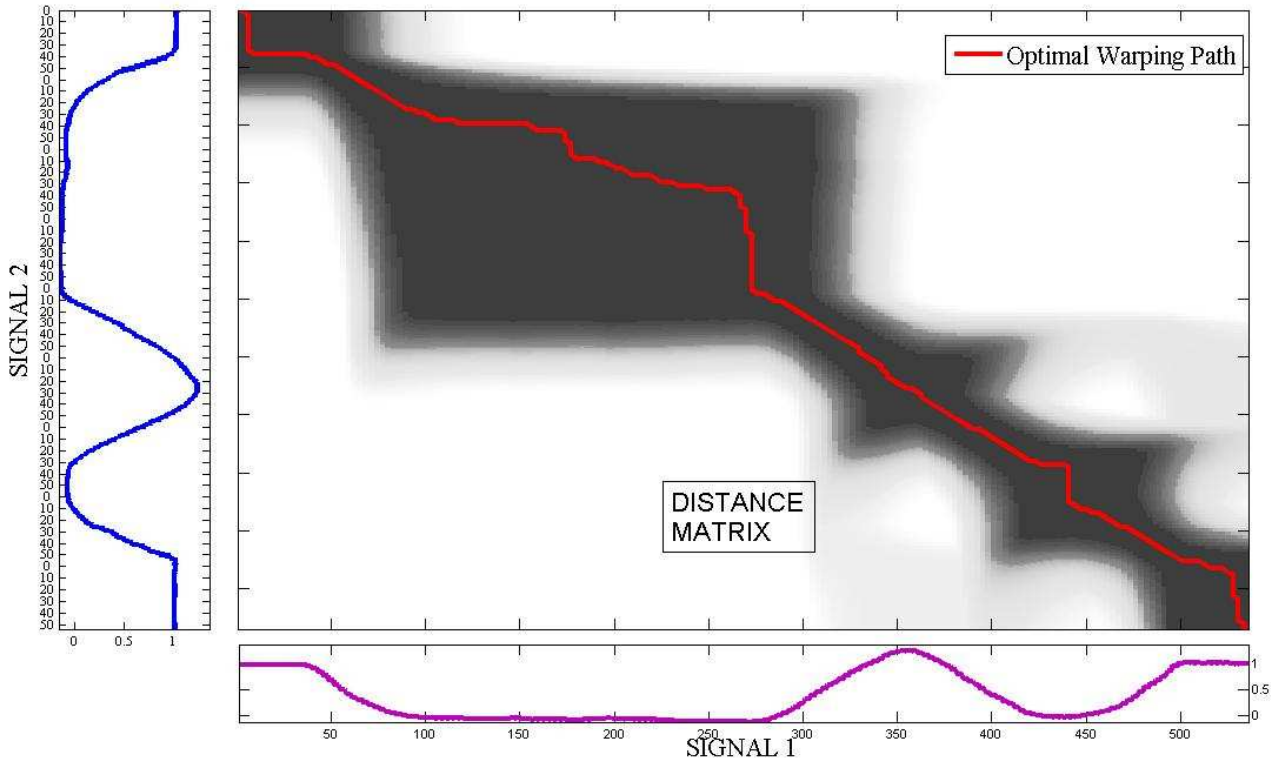


Figure 2.5: Dynamic Time Warping (DTW)

The method can be explained using Figure 2.5. Consider the two joint angle trajectories as being the two signals A and B. We calculate a distance matrix $D_{I \times J}$

whose elements are $D(i,j) = ||a_i - b_j||$. Now, the problem involves finding the least cost path from $D(1,1)$ to $D(I,J)$. This least cost path will give us the desired warping function. The least cost path can be found out using the dynamic programming algorithm.

$$p(i,j) = D(i,j) + \min\{p(i-1,j),\ p(i,j-1),\ p(i-1,j-1)\} \tag{2.7}$$

where $p(i,j)$ is the least cost path (found iteratively) of the point $D(i,j)$ from the origin $D(1,1)$.

To warp the trajectories with one another, we have to first decide a template trajectory. All the trajectories for the same joint angle for the given hand gesture are warped to this template trajectory. For selecting the template, we find the distance of the warping path for each trajectory with the others. We select that trajectory as the template, which gives the least warping distance for all other signals in the same class.

2.3.5 Weighted Averaging

Once the trajectories are optimally aligned in time, their general representation must be obtained. It can be seen from Figure 2.6 that there is very small variance between the trajectories after they are aligned. Hence, an averaging approach is sufficient. Instead of simple averaging we use weighted averaging. Intuitively, the weighing function for each element should be a function which is inversely proportional to its distance from the class mean. Thus, the elements closer to the class mean get weighed more heavily than the others. Thus, this reduces the distortions in the averaged trajectory because of possible outliers. Suppose $\{x_1, \cdots x_n\}$ are the n joint angle trajectories we want to generalize. Since each of the signals is time normalized, let T be the length of each signal. Let $\{x_{1t}, \cdots x_{nt}\}$ be the value of each trajectory at the time instance t where $t \in \{1, \cdots T\}$ We define $\mu_t = \sum_{i=1}^{n} x_{it}/n$ and ${\sigma_{it}}^2 = (x_{it} - \mu_t)^2$.

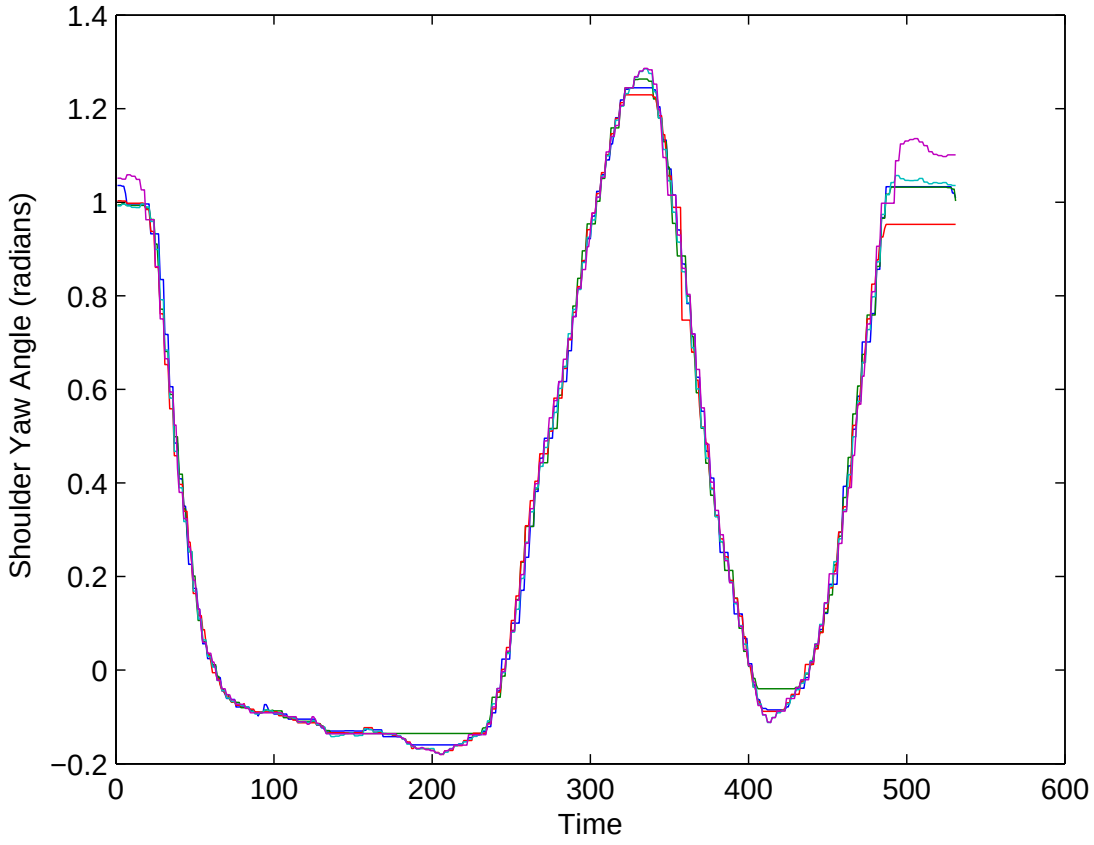


Figure 2.6: Temporally aligned trajectories for the same trajectories shown in Figure 2.4

Then the average value $\bar{x_t}$ for the time instance t is found out as,

$$\bar{x_t} = \frac{\sum_{i=1}^{n} x_{it}/{\sigma_{it}}^2}{\sum_{i=1}^{n} 1/{\sigma_{it}}^2} \tag{2.8}$$

Thus $X = \{\bar{x_1}, \cdots \bar{x_T}\}$ will be the generalized version of the n given trajectories. This trajectory is the one which the robot will emulate.

2.4 Results of Imitation Learning

The imitation learning framework was tested on common arm gestures to evaluate our system. The human subject was asked to perform a gesture in his own style for 5

trials (arbitrarily chosen). The gestures on the basis of which our system is evaluated are :

1. Sweeping motion : Simple sweeping motion. Use of Shoulder Yaw and Elbow Yaw angles are prominent.

2. Knocking motion : Motion of knocking door. Use of Elbow Yaw and Elbow Roll angles are prominent

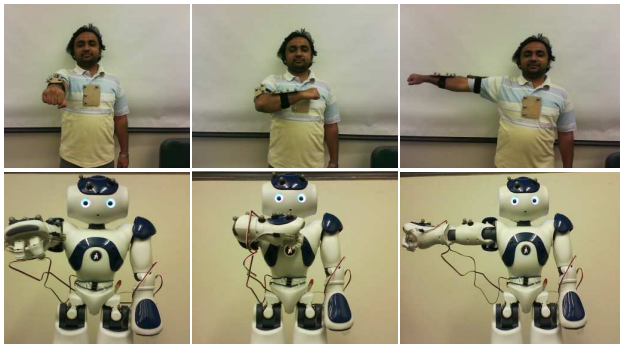
3. Writing capital letter 'N' : Simple alphabet containing straight lines

4. Writing capital letter 'D' : Alphabet containing a curve

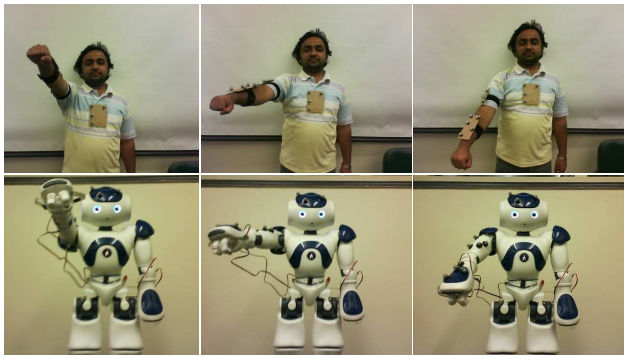
The joint angles of the subject are calculated using the data from the motion capture system. These angle trajectories are then filtered with an averaging filter to remove noise if any. Once all the trials are captured, the Dynamic Time Warping algorithm is applied to align the trajectories for the each joint angle in time. The aligned trajectories are generalized using the Weighted Averaging strategy described above. Once we obtain the generalized trajectories for each joint, we apply these trajectories to the respective joints of the robots and obtain the results. Figure 2.7 shows the robots imitating action as compared to the action performed by the human subject, at various instances The robot is able to closely imitate the hand gesture performed by the human subject

For validating the goodness of the proposed method, we attached the marker plates to the humanoid's arm in a way similar to the human being. We follow the same procedure described in section 2.3.2 to calculate the joint angle trajectories for the robot's arm. We compare the trajectory hence obtained, with the generalized trajectory for the human subject. This gives us a direct measure of how well the robot is imitating the human subject.

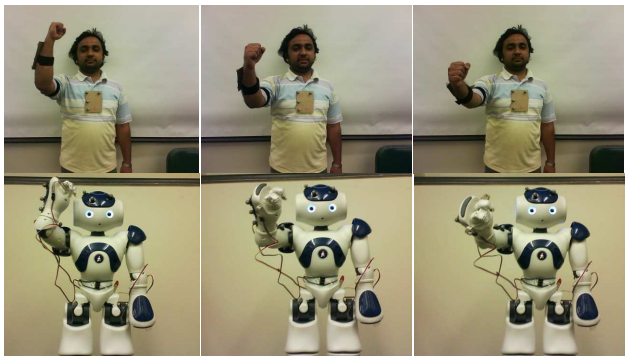
Figure 2.8 compares the generalized trajectory (shown in blue) with the trajectory of the shoulder yaw angle performed by the robot (shown in red). The mean square



(a) Sweep Motion

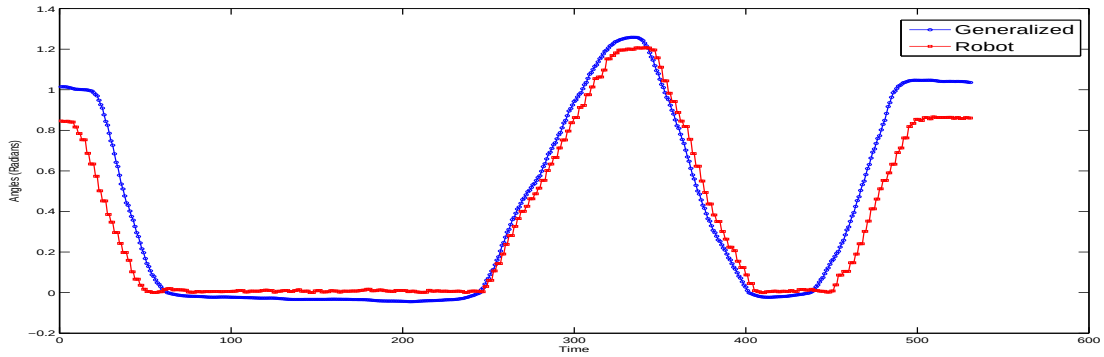


(b) Writing letter 'D'

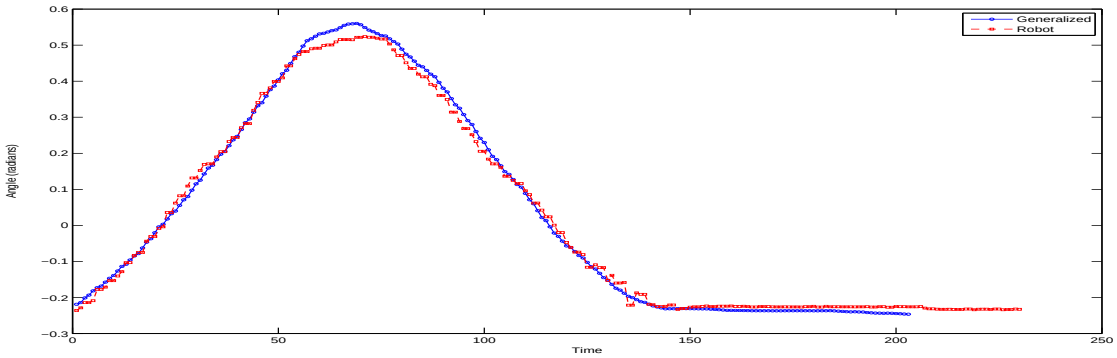


(c) Knocking

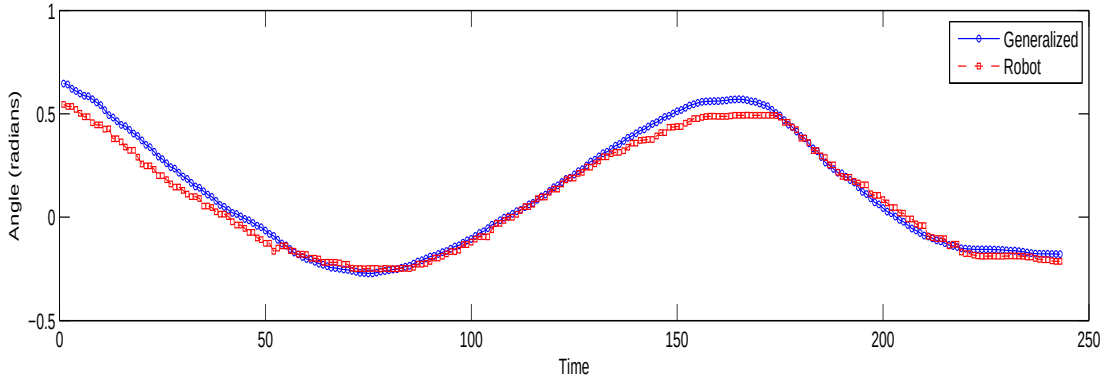
Figure 2.7: Sequence of actions for various gestures performed by subject and imitated by the robot



(a) Shoulder yaw angle for 'sweep' motion



(b) Shoulder pitch angle for writing letter 'D'



(c) Shoulder pitch angle for writing letter 'N'

Figure 2.8: Generalized trajectories(blue circles) and trajectories observed from robot(red squares)

Table 2.1: Mean Square Error

	Sweeping	**Letter 'D'**	**Letter 'N'**	**Knocking**
Shoulder Yaw	0.0189	0.0041	0.0029	0.0034
Shoulder Pitch	0.0043	0.0039	0.0052	0.0061
Elbow Yaw	0.0075	0.0243	0.0019	0.0326
Elbow Roll	0.0033	0.0075	0.0167	0.0047

error between these two trajectories is calculated and presented in table 2.1. We can observe that the performed trajectory closely matches the generalized trajectory. It can also be observed in Figure 2.8(a) that some clipping occurs, because the shoulder yaw angle for the robot ranges from 0 to $\pi/2$. Upon observing the motion of the robot in real time, there is not much distortion because of this clipping.

2.5 Missing Data in Joint Angle Imitation

We now consider the case where the demonstration data contains missing segments. Missing data problem occurs in marker based systems when the markers are obstructed or even when the cameras fail to locate the correct position of the markers because of bad calibration. The problem is more severe in marker-less systems using computer vision, where missing data may occur due to occlusions, insufficient lighting and general algorithmic failures. Hence, it is necessary to address this problem.

Although multiple demonstrations of the same gesture are available, the problem of filling in the missing gaps is not straight-forward since the demonstrations are not time-aligned. The problem is worsened by the fact that the time alignment operation involves non-linear time shifting and scaling operations. Thus, fundamentally, we have to derive a time-alignment function by performing non linear optimization so that the difference between the aligned multiple demonstrations is minimized in some sense.

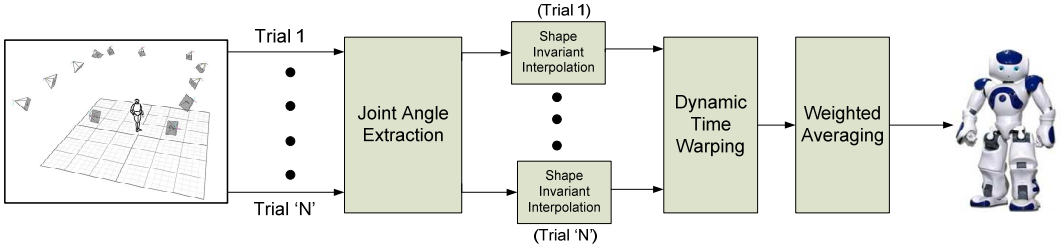


Figure 2.9: Interpolation approach block diagram

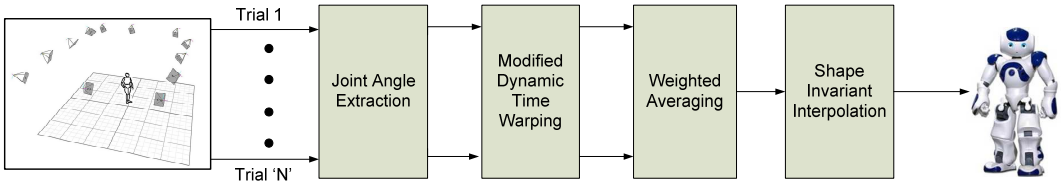


Figure 2.10: Modified DTW approach block diagram

In this section, we propose and evaluate two approaches to solve the missing data problem. One is the interpolation approach and the other approach uses a modified version of the Dynamic Time Warping algorithm [11].

Figures 2.9 and 2.10 show the overall system design for the two proposed approaches. Data is collected from the motion capture system. The joint angle trajectories are extracted from the motion capture data which has missing data. Figure 2.9 shows the interpolation approach to solving the missing data problem. Figure 2.10 shows the block diagram of the modified DTW approach.

For a given signal or trajectory, there can be two sources of information to infer the missing data. Horizontally, we can infer the missing data by interpolating across the trajectory. Vertically, we can infer missing data by guessing the correspondence between the trajectories arising from multiple demonstrations. But, before we can infer data vertically, the trajectories must be time-aligned. Hence, we can have two approaches. In the interpolation approach, we first interpolate along each individual trajectory and then apply time warping and generalize by weighted averaging. In the modified DTW approach, we apply the DTW algorithm to the fragmented trajectories and then generalize. The generalized trajectory in the latter case can still contain

missing data which can be filled up by interpolation. Thus, in both the proposed approaches, we make use of the information contained vertically and horizontally.

2.5.1 Interpolation approach

In this approach, the missing raw joint angle trajectory data is reconstructed independently through interpolation. Interpolation is done using a piecewise cubic hermite interpolant as described in [20]. The advantage of this technique is that it generates shape preserving curve which is not possible using cubic splines. Practically, it was observed that using this interpolant, best reconstruction results for joint angle trajectories are possible. Once all the trajectories are reconstructed, we apply Dynamic Time Warping to achieve temporal alignment. After the trajectories are aligned, weighted averaging is applied to obtain a generalized trajectory for all the demonstrations.

2.5.2 Modified DTW approach

The DTW algorithm cannot handle signals with missing data. Since the given signals do not have continuity, the cost matrix $d(i, j)$ cannot be computed. Hence, some adjustments are needed. For the sake of constructing the cost matrix, we apply the piecewise hermite interpolant to the given signals. A record of the time steps which had to be interpolated is kept. Once the cost matrix is constructed, the DTW algorithm proceeds as usual. After a shortest path has been found out, the parts of the path which correspond to the interpolated time steps of the original signals are removed. The reasoning behind doing this is that the interpolated data points should have no role in producing the final generalized output. Hence, we are left with a broken piecewise time warping function as shown in Figure 2.11.

Further, this fragmented time warping function is interpolated using the shape invariant interpolant. Equivalently, this corresponds to predicting the warping for

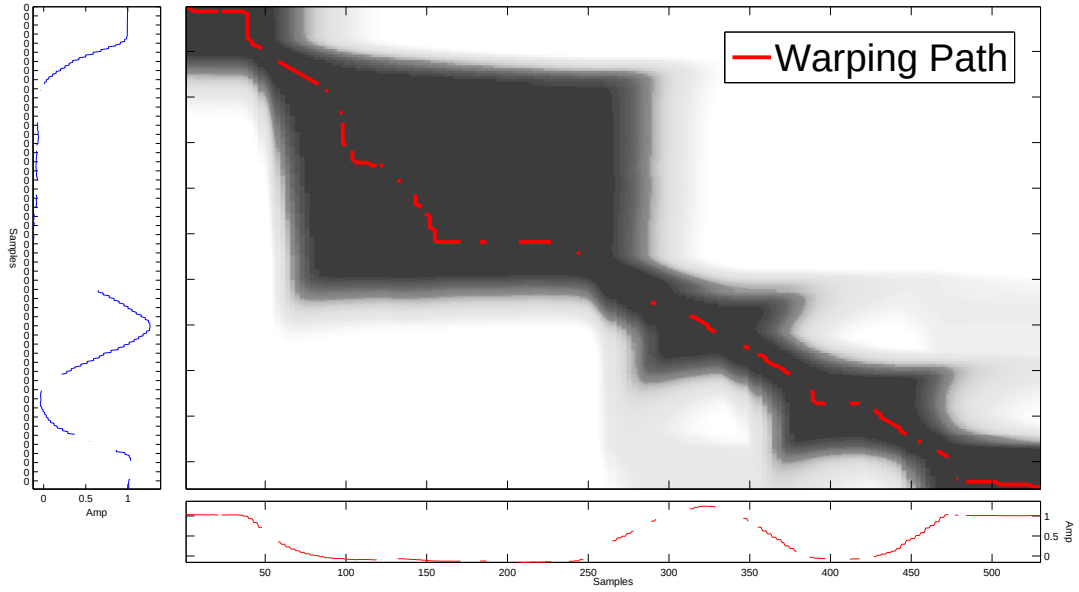


Figure 2.11: Broken warping path obtained by modified DTW approach

the missing parts of the signals. This is shown in Figure 2.12. This approach is different from the interpolation approach, because in this approach, interpolation on the signals is used temporarily to construct the cost matrix.

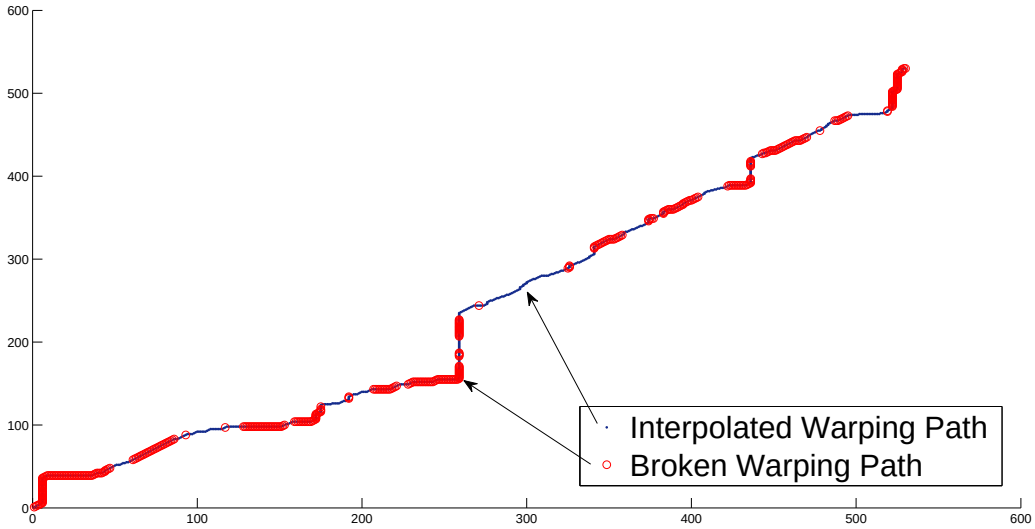


Figure 2.12: Interpolating the broken warping path using a shape preserving interpolant

2.5.3 Results of both approaches

The algorithms again tested on the arm gestures such as sweep motion, knock motion and writing alphabets 'D' and 'N'. First, we obtain the data from the motion capture system and compute the joint angles of the arm. To simulate the missing data conditions, patches of specified length (time steps) are randomly removed until a specified percentage of data is lost. As in our previous work [21], we apply DTW and weighted averaging to the original motion capture data and produce generalized trajectories. We consider these as the ground truth. We present the results of the interpolation and modified DTW algorithms by comparison with ground truth.

As a distance metric for comparing with ground truth, Euclidean distance does not perform well. We use the DTW algorithm to find the distance of the generalized results from the ground truth. Tables 2.2 and 2.3 show the comparison of error obtained by using the two approaches for the various gestures. For each gesture, 5 demonstrations were considered for training and the indicated amount and length of missing data was added. Experiments were performed by varying the amount and length of missing data. The amount of missing data (%) indicates the percentage of data missing from the trajectory. The length of missing data indicates the size of the biggest gap (in time steps). The last two columns show the error obtained by the corresponding approaches as DTW distance with the ground truth. The general observation is that, as the percentage of missing data increases, the error (DTW distance) increases. It is also seen that for the same percentage of missing data, the length of the missing data impacts the error drastically. The reason for this is, the interpolation becomes poor for longer gaps. Also, the error is more for complicated hand gesture like 'knock' as compared to the other gestures. The results are acceptable when error is less than 5 rad^2. Upon comparing the results obtained by the two algorithms, we find that the Interpolation approach performs better than the Modified DTW approach. The reason might be, because we remove the interpolated

Table 2.2: Comparison of approaches to sweep and knock gestures

Gesture	Length missing (time steps)	Percent missing (%)	Interpolation approach (rad^2)	M-DTW approach (rad^2)
Sweep	10	20	0.0442	0.2511
		40	0.0856	0.4300
		60	0.1334	0.8920
	20	20	0.0731	0.2435
		40	0.1122	0.4731
		60	0.5293	1.4773
	50	20	0.2257	0.4146
		40	0.6218	0.5153
		60	2.9137	9.3397
Knock	10	20	0.4754	1.0993
		40	0.6517	3.8922
		60	0.6854	3.2733
	20	20	0.5826	1.1051
		40	1.2573	7.0438
		60	1.8742	13.738
	50	20	1.3201	2.3362
		30	2.4421	6.5206
		40	11.431	15.512

Table 2.3: Comparison of both approaches to tracing 'N' and 'D'

Gesture	Length missing (time steps)	Percent missing (%)	Interpolation approach (rad^2)	M-DTW approach (rad^2)
'N'	10	20	0.0154	0.0262
		40	0.0233	0.0356
		60	0.0491	0.0736
	20	20	0.0191	0.1952
		40	0.0385	0.2426
		60	0.0837	1.4276
	50	20	0.1000	0.2640
		40	0.5861	0.9860
		60	2.8497	3.9402
'D'	10	20	0.0094	0.0581
		40	0.0211	0.0725
		60	0.0343	0.0896
	20	20	0.0066	0.0521
		40	0.0561	0.0822
		60	0.4326	0.1983
	50	20	0.0216	0.1643
		40	0.0333	0.2112
		60	2.9372	4.2631

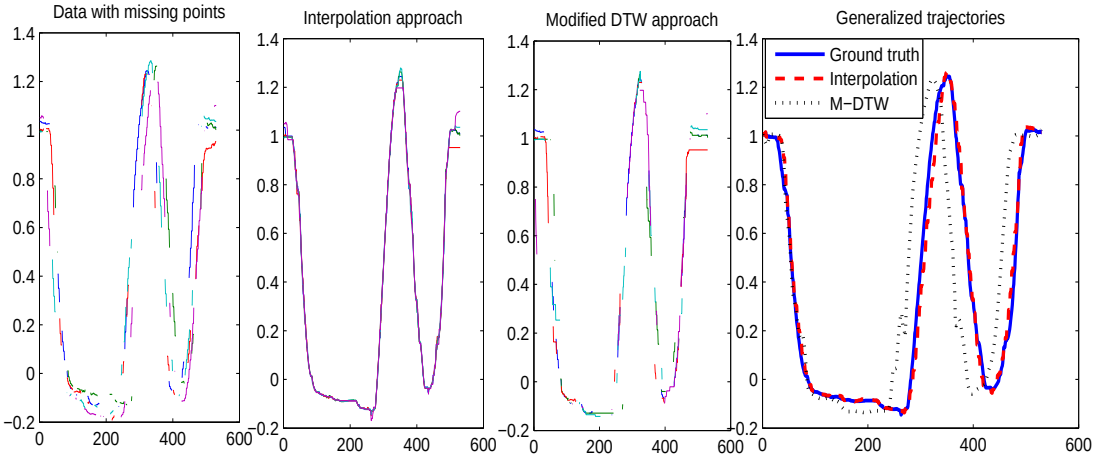


Figure 2.13: Results for shoulder yaw angle for learning the 'sweep' gesture

points in the M-DTW approach, the system falls short of data required for learning. The M-DTW approach truly does not infer the trajectory information contained horizontally.

The results for the shoulder yaw angle trajectory during the 'sweep' gesture are shown in Figure 2.13. The figure shows the raw data with missing points, the results obtained by interpolation and modified DTW approaches, and the generalized trajectories which the robot follows. The ground truth is obtained by applying DTW to the original data (without missing points) and generalizing it. Figure 2.14 shows the trajectories followed by the tip of the robot's hand during the gestures of 'Knock', 'D' and 'Sweep'.

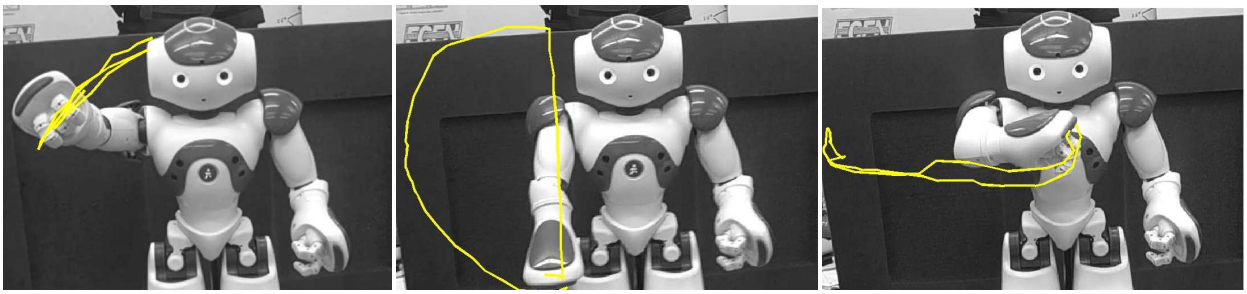


Figure 2.14: Trajectories of 'Knock', 'D' and 'Sweep' gestures

This section discussed how the missing data problem can be alleviated by using the interpolation and the M-DTW approaches. Essentially, the M-DTW approach involves removing the artifacts obtained from the interpolation step. The results show that the interpolation approach still performs better than the M-DTW approach. Although the artifacts occuring from interpolation are removed in the M-DTW approach, the artifacts occuring because of warping are not removed. Artifacts appear after warping since we use the warping path directly to manipulate the signals. It can be seen from Figure 2.12 that the warping path is quite rough, which leads to abrupt signal warping. Instead of interpolating the warping path with a shape invariant interpolant in the M-DTW approach, a smoother interpolant might lead to better results. Further, adding slope constraints to the warping path might also help.

In this chapter, we have seen how robots can be taught to perform simple arm gestures by imitation learning. The learnt arm gestures can be used extensively while interacting with humans. Apart from arm gestures, robots can also be taught to perform various tasks using the learning from demonstration paradigm. We discuss this in the next chapter.

CHAPTER 3

LEARNING TASKS FROM DEMONSTRATIONS

In the previous chapter, we have seen how robots can be taught to perform simple arm gestures by imitation learning. It would be highly desirable to transfer visually guided skills such as grabbing an object, from the human to the robot using demonstrations. This concept is vastly applicable in personal service robots, where the end-users would be non-experts having limited programming knowledge. The idea of teaching tasks by showing, would give them a very intuitive programming capability, and enable them to teach robots new tasks. In our work, we consider a task of teaching the robot to grasp a table at a specific point.

However, transferring task skills is more challenging than transferring imitation skills. This is because, the task environment need not be exactly the same as when it was demonstrated. For example in the table grasping task, the position of the table could be different than when the task was demonstrated. Hence, the robot cannot reproduce the task by simply recording human's joint angles and replaying them. For the robot to actually learn the task, it has to observe motion of the human's hand with respect to the object and extract the essential constraints. Again, the problems involved are data representation, generalization and reproduction.

3.1 Related Work

Amongst the robot learning from demonstration paradigm, there are several questions that determine the structure of the learning framework. These questions are *what to imitate, how to imitate, when to imitate* and *whom to imitate.* A large body of

research tries to answer the first two questions at different levels.

Based on the distinction made, there exist three major approaches -

- Low level approaches - where the robot observes and learns raw visuo-motor skills (answers the *how to imitate* question)
- Mid level approaches - answers the *what to imitate* and *how to imitate* questions
- High level approaches - where the robot has a semantic understanding of the world and tries to observe and learn tasks symbolically (answers the *what to imitate* question)

The last chapter was a good example of a lower level approach, where the robot learns motion primitives in the joint space. However, such lower level learning is limited in producing only simple robot behaviors. In higher level approaches, it is assumed that the robot has some prior knowledge about motion primitives. For example, in [22], the robot is taught a task to lay table for dinner. Typically, in such approaches, tasks are split into subtasks or actions, and their interdependence and sequencing is learnt. The learning could result into the robot building up an abstract knowledge base such as *Pickup Bowl, Place Bowl, Pickup Saucer* and so on. In such approaches, the prior knowledge assumed is that the robot knows *how to pickup the bowl*.

Mid level approaches have started gaining more popularity, since they answer both the *what to imitate* and *how to imitate* questions simultaneously. Mid level approaches extract the essential task constraints from the given demonstration. These constraints are with respect to the task environment. Given a new state of the task environment, mid level approaches determine the new constraints (what to imitate) and use the existing demonstrations to derive controllers for task reproduction (how to imitate). However they require a larger number of human demonstrations as compared to higher level approaches.In this work, we use a mid level approach which uses the

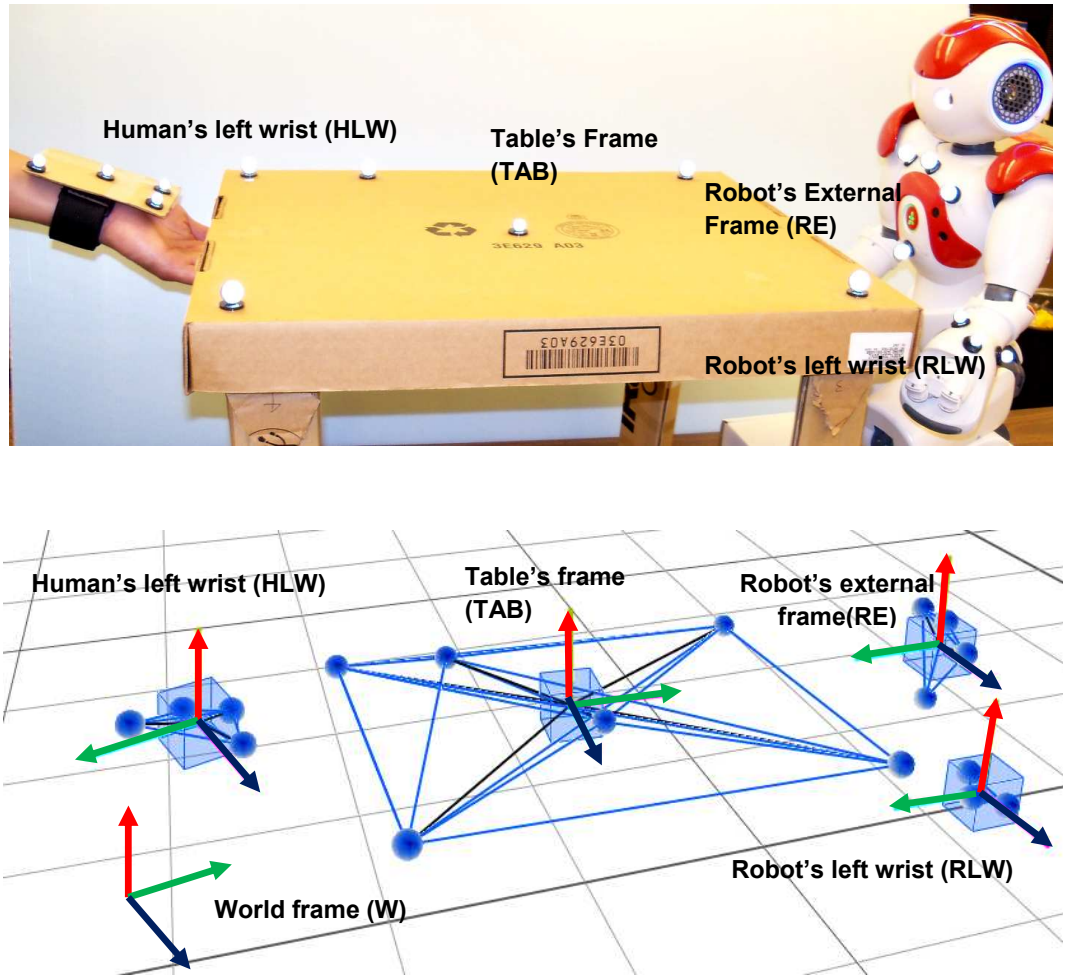


Figure 3.1: Experimental setup and the various co-ordinate frames involved

GMM/GMR framework proposed by Calinon *et al.* [23]. The framework provides a continuous representation of the task constraints which can be used to generalize and reproduce the gestures.

3.2 Experimental Setup

The experimental setup mainly consists of the motion capture system and the humanoid robot. Figure 3.1 shows the experimental setup. C++ is used at the front-end for communicating with the robot and MATLAB® is used at the back-end for processing data. Markers are attached only to the human's wrist to track his hand position in the work-space. Markers are placed on the table as well as the robot.

Table 3.1: Co-ordinate frames involved

Rigid Body	Notation
Human left wrist	HLW
Robot left wrist	RLW
Table	TAB
Robot's external frame (torso)	RE
World frame	W

As we had seen in the earlier chapter, the motion capture system allows us to create 'rigid body' objects. Each rigid body defines its own co-ordinate frame having its translation and rotation with respect to the global frame. The rigid body frames defined for our setup are listed in Table 3.1. The convention we follow in this paper are: the X-Y-Z co-ordinates (position) of a rigid body 'A' with respect to rigid body 'B' is denoted as ${}^{A}P_{B}$ and the rotation of body 'A' with respect to body 'B' is denoted by ${}^{A}R_{B}$

Firstly, the human demonstrates the table-reaching action multiple times, which is captured by the motion capture system. The goal is to extract the constraints from the demonstration, and map them to the robot embodiment. However, the end effector position of the robot is controlled by a frame located somewhere inside its torso. We shall call this the robot's 'internal' frame denoted by RI. The table's frame is used to observe the human's hand motion with respect to table during the demonstration. The human demonstrations have to be mapped to the robot by removing the embodiment difference that exist between the human and the robot. This mapping is simplified by considering all the trajectories in the *task space* as opposed to the joint space. Calibration is further needed to derive the transformation for converting the trajectories in robot's external frame to the robot's internal frame. Calibration is discussed further in this section.

3.2.1 Calibration

The robot's end-effector has to be controlled with respect to its internal frame of reference. But the mapped data obtained from demonstrations, is the trajectory of the markers placed on the robot's hands with respect to the markers placed on the robot's torso. Hence, a calibration is needed to establish a correspondence between the external marker frame with the internal robot frame. The robot's SDK can provide the position of the robot's end effector with respect to its internal frame of reference. Hence given the corresponding motion capture data and encoder data, a transformation can be derived. We model this transformation as a homogenous transformation which takes care of scaling, translation and rotation.

For calibration, the robot waves its hand in random trajectories trying to cover all the possible joint configurations of its arms. While it is doing so, positions are collected simultaneously from motion capture (denoted by A) and forward kinematics applied to robot's internal joint encoders (denoted by B). The linear least squares formula used to calculate this homogenous transformation (H) can be given as

$$H = (A^T A)^{-1} A^T B \tag{3.1}$$

It can be easily seen that $B = AH$ would convert the motion capture trajectory (A) to the trajectory that can be enacted by the robot.

3.3 Methodology

For learning the task from demonstrations, we adopt the probabilistic learning framework proposed by Calinon *et al* [24]. Firstly, a Gaussian Mixture Model (GMM) is used to encode the set of demonstrated trajectories (data-representation problem). Then, Gaussian Mixture Regression (GMR) is applied to the GMM model to retrieve a smooth generalized version of these trajectories and associated variances (general-

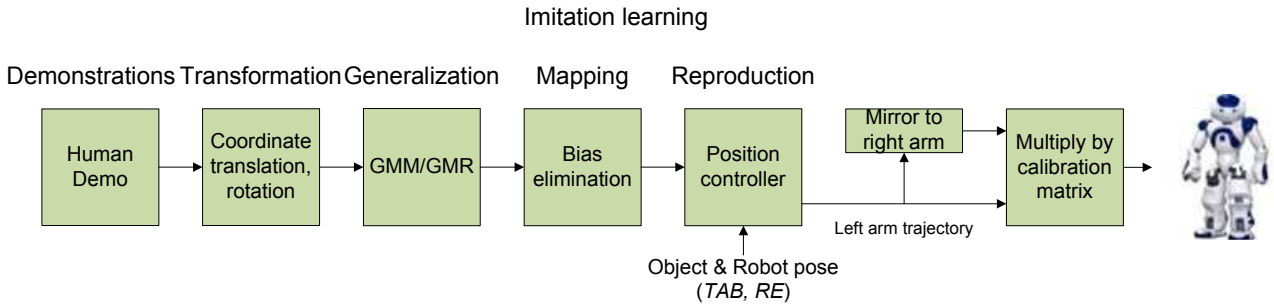


Figure 3.2: Learning tasks from demonstration framework

ization problem)[25]. These generalized trajectories are then mapped to the robot's embodiment (human-robot mapping problem).

In a new scenario, such as new position and orientation of the robot and the table, a position controller is derived from the generalized trajectories (reproduction problem). The block diagram for the learning tasks from demonstration framework is shown in Figure 3.2. The details of the block diagram are described next.

Coordinate transformation

Various coordinate transformations are required for converting the captured trajectories to the trajectories of actual interest. The obtainable trajectories from the motion capture are ${}^{HLW}P_W$, ${}^{REF}P_W$ and ${}^{TAB}P_W$ which are with respect to the world frame. The *human's wrist trajectory with respect to the table* (denoted by ${}^{HLW}P_{TAB}$) is of interest for learning. So we need to transform the trajectory ${}^{HLW}P_W$ from the world frame to the table's frame given the pose of the table object (TAB). The transformation essentially is first a translation and then a rotation. It is given by

$$ {}^{HLW}P_{TAB} = {}^{TAB}R_W({}^{HLW}P_W - {}^{TAB}P_W) \tag{3.2}$$

Generalization

Let $\{\varepsilon_j\}_{j=1}^{N}$ denote the N demonstrations. Each demonstration is normalized to 100 time steps. Each datapoint $\varepsilon_j = \left\{t_j, \varepsilon_j^S\right\}$ consists of a time step t_j and a coordinate of position ε_j^S which is a point in trajectory of the human's left wrist with respect to the table, ${}^{HLW}P_{TAB}$. The dataset is first modeled by a Gaussian Mixture Model(GMM) of K components, each data point is defined by its probability density function[26]

$$p(\epsilon_j) = \sum_{k=1}^{K} \pi_k N(\epsilon_j; \mu_k, \Sigma_k) \tag{3.3}$$

where, π_k are prior probabilities and $N(\epsilon_j; \mu_k, \Sigma_k)$ are Gaussian distributions defined by centers μ_k and covariance matrices Σ_k, whose temporal and spatial components can be represented separately as

$$\mu_k = (\mu_k^T, \mu_k^S), \;\; \Sigma_k = \begin{pmatrix} \Sigma_j^{TT} & \Sigma_j^{TS} \\ \Sigma_j^{ST} & \Sigma_j^{SS} \end{pmatrix} \tag{3.4}$$

Based on the GMM, a generalized version of the trajectories is computed by applying Gaussian Mixture Regression (GMR). The procedure is as follows. For each component k, the expected distribution of likelihood of ε_j^S given a time step t_j and gaussian mixture component k is defined by

$$p(\epsilon_j^S | t_j, k) = N(\epsilon_j^S;\ \hat{\epsilon}_k^S, \hat{\Sigma}_k^{SS}) \tag{3.5}$$

$$\hat{\epsilon}_k^S = \mu_k^S + \Sigma_k^{ST}(\Sigma_k^{TT})^{-1}(t_j - \mu_k^T) \tag{3.6}$$

$$\hat{\Sigma}_k^{SS} = \Sigma_k^{SS} - \Sigma_k^{ST}(\Sigma_k^{TT})^{-1}\Sigma_k^{TS} \tag{3.7}$$

By taking the complete GMM into account, the expected distribution is defined by

$$p(\epsilon_j^S | t_j) = \sum_{k=1}^{K} \beta_{k,j} N(\epsilon_j^S;\ \hat{\epsilon}_k^S, \hat{\Sigma}_k^{SS}) \tag{3.8}$$

Figure 3.3: Embodiment difference between the human's hand and the robot's hand.

where $\beta_{k,j}$ is the probability of the component k responsible for t_j. By using the linear transformation property of Gaussian distribution, and estimation of the conditional expectation of ϵ_j^S given t_j is thus defined by $p(\epsilon_j^S \ | t_j) \ \propto \ N(\hat{\epsilon}_j^S, \hat{\Sigma}_j^{SS})$, where the parameters of the Gaussian distribution are defined by

$$\hat{\epsilon}_j^S = \sum_{k=1}^{K} \beta_{k,j} \hat{\epsilon}_k^S, \ \ \hat{\Sigma}_j^{SS} = \sum_{k=1}^{K} \beta_{k,j}^2 \hat{\Sigma}_k^{SS} \tag{3.9}$$

By evaluating $\left\{\hat{\epsilon}_j^S, \hat{\Sigma}_j^{SS}\right\}$ at different time steps t_j, a generalized form of the trajectories $\hat{\epsilon} = \left\{t_j, \hat{\epsilon}_j^S\right\}$ and associated covariance matrices $\hat{\Sigma} = \left\{\hat{\Sigma}_j^{SS}\right\}$ representing the constraints along the task can be computed [23].

Correspondence problem

The constraints are derived from ${}^{HLW}P_{TAB}$ which are obtained from the human demonstrations. The constraints derived for ${}^{HLW}P_{TAB}$ have to be mapped to the robot's end effector with respect to the table denoted by ${}^{RLW}P_{TAB}$. All the trajectories considered for learning are in the task space as opposed to joint space. Hence, inherently the embodiment mapping problem is simplified. The mapping problem is further simplified by mapping only the positional constraints. Hence, only the bias difference shown in Figure 3.3 between the human's wrist and the robot's end effector has to be taken care of. A simple method is proposed to calculate this dimension

difference. The human and robot grasp a fixed object in space. The coordinates with respect to the fixed object are obtained. The difference between these two coordinates is the required bias compensation. Hence, a simple human to robot mapping can be achieved.

Reproduction

In the reproduction phase, a new trajectory for the robot's end effector ${}^{RLW}P_{RE}$ has to be produced based on the generalized version of ${}^{RLW}P_{TAB}$. Given the pose of the table (TAB) during reproduction phase, ${}^{RLW}P_{RE}$ can be derived as follows:

We have ${}^{RLW}P_{TAB}$ which is

$$ {}^{RLW}P_{TAB} = {}^{TAB}R_W({}^{RLW}P_W - {}^{TAB}P_W) \tag{3.10} $$

${}^{RLW}P_W$ can be obtained as

$$ {}^{RLW}P_W = {}^{TAB}R_W{}^{-1}\ {}^{RLW}P_{TAB} + {}^{TAB}P_W \tag{3.11} $$

Finally we can derive ${}^{RLW}P_{RE}$ as

$$ {}^{RLW}P_{RE} = {}^{RE}R_W({}^{RLW}P_W - {}^{RE}P_W) \tag{3.12} $$

${}^{RLW}P_{RE}$ is then transformed for by calibration to yield a trajectory that can be enacted by the robot.

Mirroring the trajectory

In the imitation learning phase, only left hand demonstrations are provided to the robot. This trajectory is mirrored across the vertical robot axis to obtain the corresponding trajectory for the robot's right end effector.

3.4.1 Calibration results

The calibration matrix is obtained by moving the robot arm randomly covering as many configurations as possible, during which the coordinates of the robot's left arm with respect to its own torso is collected both in the motion capture and the robot's internal frame.

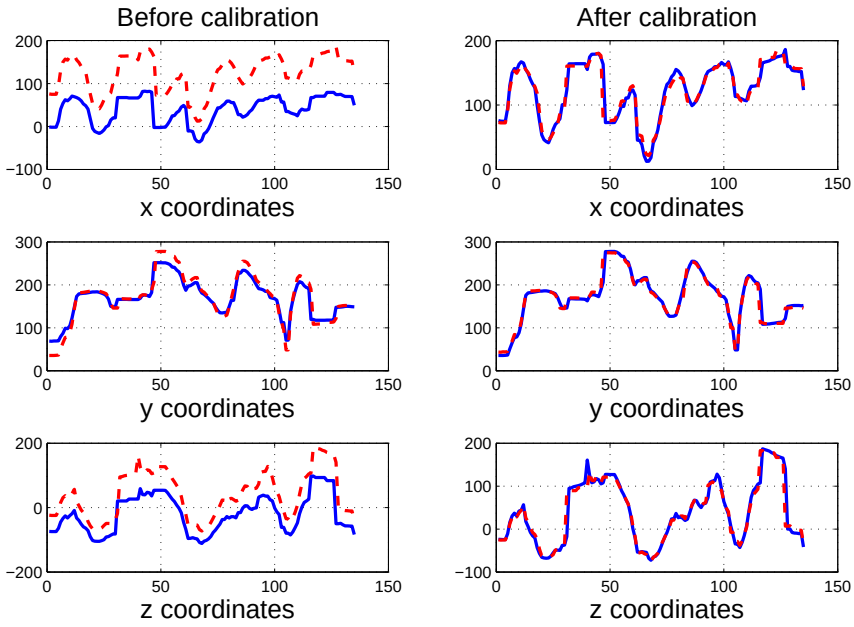


Figure 3.4: Calibration Results

Then, the homogeneous transformation is used to calculate the calibration matrix. The homogenous matrix basically gives a transformation for the robot hand motion from motion-capture to fit the data obtained from the internal encoder data. Figure 3.4 shows the coordinates in the two frames before and after calibration. It can be observed from the figure, that using the calibration matrix the trajectories can be successfully converted from the motion capture frame to the robots internal frame.

3.4.2 Imitation learning results

In the imitation learning phase, multiple demonstrations are performed by the human. In each demonstration, the human tries to approach the same position of the table with his left hand from an arbitrary initial position. An open source MATLAB code has been used to implement GMM/GMR [23]. The GMM/GMR results are shown in Figure 3.5. Generalized trajectories and constraints are thus obtained.

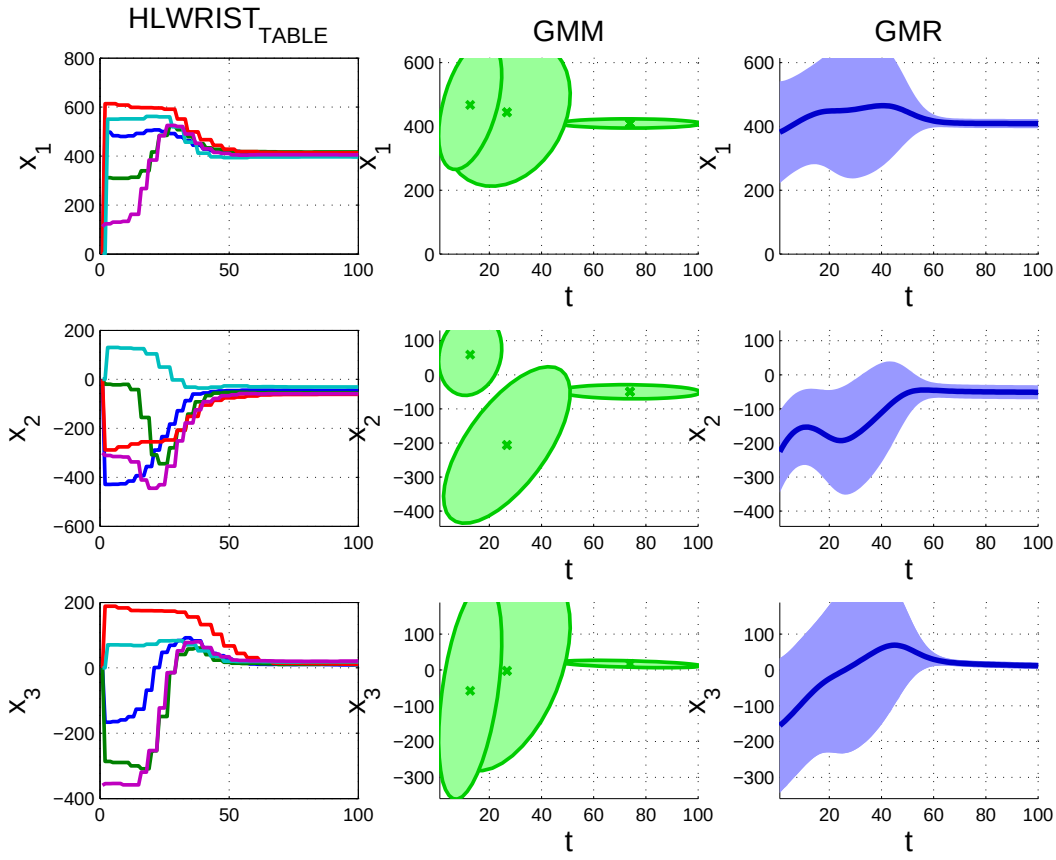


Figure 3.5: Trajectory encoding and generalization.

From the results, it can be seen that the constraints of the human hand's final position with respect to the table becomes narrower, which suggests that the final position of the robot's end effector with respect to the table should also be consistent. After compensating for the bias difference between human's hand and robot's hand, the robot can generate its own trajectory given the extracted constraints.

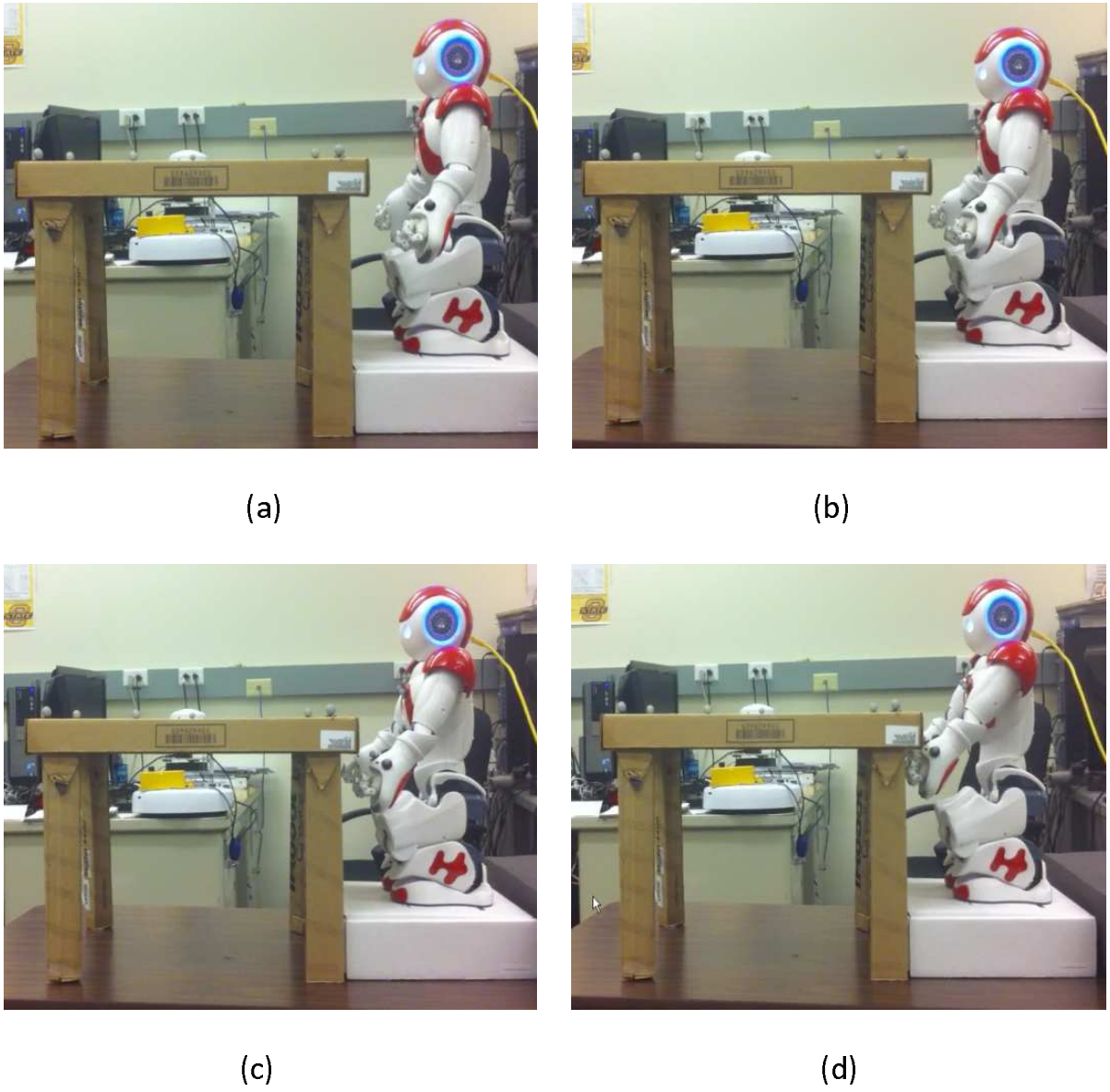


Figure 3.6: Replaying the generalized trajectory. (a)-(d)

Given a new position of the table with respect to the robot, a new trajectory is reproduced by the position controller. The calibration matrix is then used to convert the trajectories from the robot's external frame to the robot's internal frame. In the imitation learning phase, demonstrations of grasping the table with only the left hand are provided to the robot. This trajectory is mirrored to obtain the corresponding

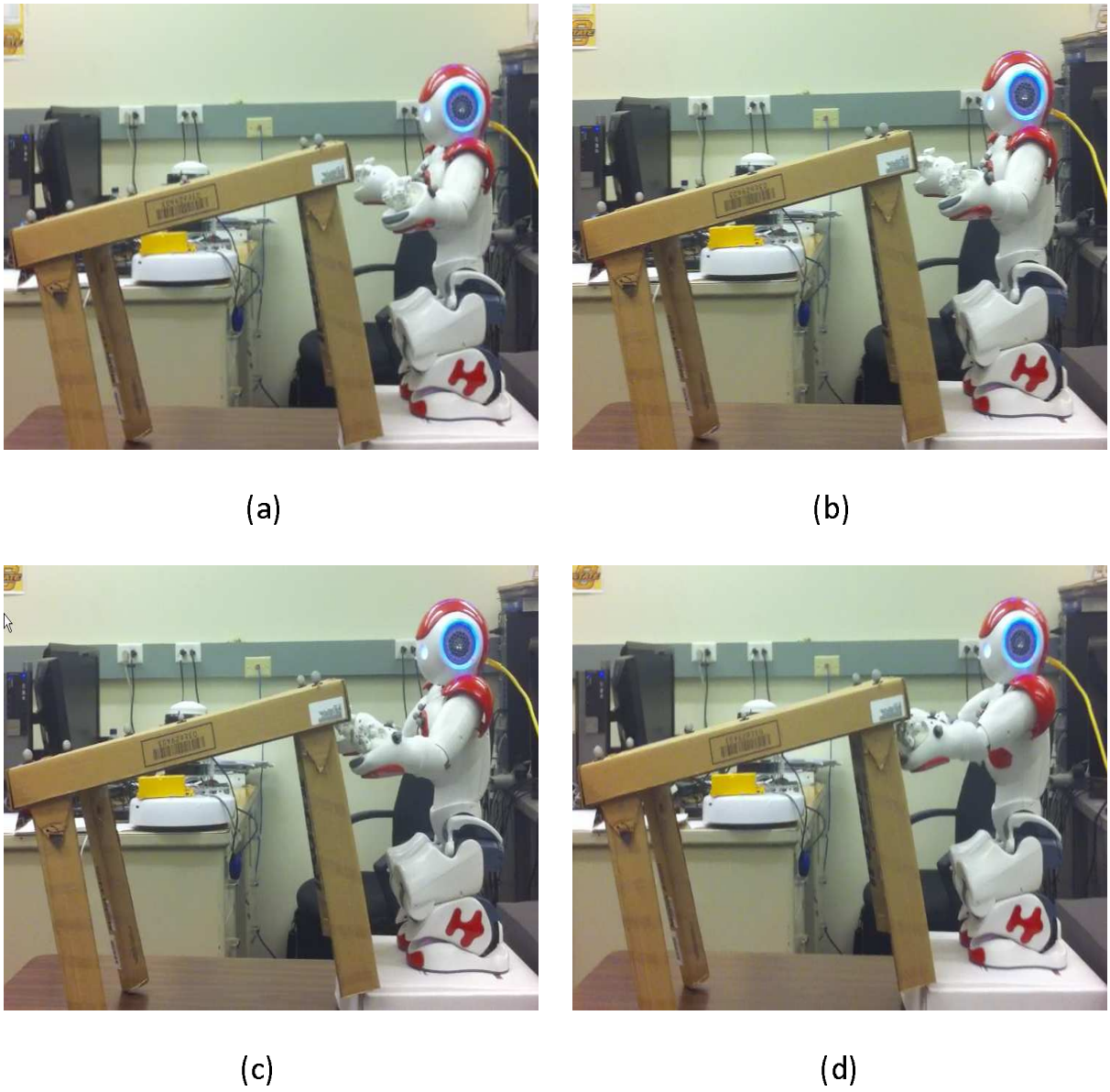


Figure 3.7: Reproducing the generalized trajectory in an unknown position. (a)-(d)

trajectory of robot's right end effector. The results are shown in Figure 3.6 and Figure 3.7. The former is the robot replaying the generalized trajectories extracted from the demonstrations and the latter is the robot reproducing the trajectories in a new situation (different position of the table). It is observed that the trajectories generated are quite smooth.

Thus, in this chapter we have seen how a humanoid robot can be taught to perform a visuo-motor task by showing human demonstrations. Learning from demonstrations is a classic human-robot interaction example where gestures or tasks are communicated explicitly to the robot. Performing collaborative tasks is a good example where humans communicate implicity. In the next two chapters, we shall see how humans can convey their intentions implicitly by using their motion or simply performing actions in human-robot collaborative tasks

CHAPTER 4

IMPLICIT COMMUNICATION BY MOTION IN A PHYSICAL HRI TASK

A physical HRI (pHRI) task involves any task in which the human and robot are coupled to an object. Possibly, the best example of a pHRI task is a joint table lifting task. Other examples of pHRI tasks include shaking hands and handing over objects. Such tasks are also called *co-operative manipulation* tasks. According to the survey by Reed *et al.* [9], humans typically do not tend to communicate explicitly when performing cooperative manipulation tasks, but use a variety of implicit cues to express and understand their intentions. Our interest is to investigate how robots can utilize the cues hidden in human motion and actions to infer human intent and thus carry out the cooperative task successfully.

An important factor that shapes human-robot collaboration is the role of the robot. In collaborative tasks, agents can assume leader, follower or mixed roles. Traditionally the role of the robot has to be pre-determined. However, humans performing collaborative tasks can switch between or share the leader-follower roles effortlessly even in the absence of audio-visual cues. In the absence of explicit communication modalities, humans communicate implicitly using their motion or by performing certain actions [9]. We propose a framework that can endow robots with a similar capability. The behavior of the robot is controlled by two types of controllers such as reactive and proactive controllers. The reactive controller causes the robot to behave as a follower and the proactive controller causes it to behave as a leader. The proactive controller suggests proactive actions based on human motion prediction. The

framework relies on a novel technique to compute a measure of confidence for the prediction. This confidence measure determines the leader/follower role of the robot. Hence, the robot can switch roles during the task autonomously and dynamically. A table-lifting task which is essentially a cooperative manipulation task is considered to demonstrate the proposed framework. Performance of the human-robot team carrying out this task is experimentally evaluated.

4.1 Related Work

In earlier works, the intellectual responsibility of planning and guiding the co-operative task is placed entirely on the human while the collaborating robot is assigned a mere follower role. These robot followers are pre-programmed with simple reactive behaviors. For example, a popular approach for accomplishing a cooperative object manipulation task, is using impedance control [27], [28]. However adopting such a naive strategy requires the human to spend extra energy in dragging the robot, apart from the energy spent in moving the load itself. Furthermore, a goal such as keeping the table exactly horizontal throughout the table-lifting task is very difficult to achieve using this technique alone.

Maeda *et al.* were amongst the earliest to provide a solution to this problem, by using a human motion prediction technique which enables the robot partner to work *proactively* with the human [29]. Human motion prediction was obtained by following the assumption, that the fellow human's motion satisfies the minimum jerk model [30] in the cooperative manipulation setting. Based on estimation of the minimum jerk model parameters, the robot could predict the velocity profile of the human's motion, which could then be used to take a proactive action. This strategy was shown to reduce the human's effort for the cooperative manipulation task. Recently, we have seen a resurgence in the studies of physical human robot interaction which make use of motion prediction strategies. Corteville *et al.* presented a robot assistant which

could predict the humans motion using a Kalman filter (KF) [31]. The KF was designed according to the minimum jerk model. The amount of assistance provided by the robot throughout the entire task had to be decided beforehand. In [32] the authors proposed a solution to change the role of the robot during the task execution using a homotopy switching model, although manually. Automatic adjustment of the homotopy variable α_i which decides the role of the robot was left as an open question, for which the proposed work offers a solution. Another shortcoming in [31] and [32] is the assumption that the robot should know the destination of the object being transported so that a plan of motion could be generated. If the destination is changed mid-way, a new subtask has to be generated on the fly which is non-trivial and is a separate work in itself [33]. Apart from cooperative tasks, human motion prediction has also been applied extensively in robotic teleoperation tasks [34], [35], [36].

Recent works show that the minimum jerk model may not be suitable for cooperative manipulation tasks [37]. The minimum jerk model assumption fails when there are large perturbations in the motion trajectory, or if the human decides to change the course of the trajectory during the task execution. In such cases, the robot might fail to comply with the human, which may lead to disastrous consequences. Also, in order to apply the minimum jerk model successfully, the final position of the object must be known both to the human and robot which is cumbersome in real world situations. It is interesting to note that two humans can excel in a table-lifting task even if one does not know the final position of the object. Other related work include [38] which proposes a task-model learning approach combined with an adaptive control system. After going through a two-step learning process, the robot can work collaboratively with the human while inferring his intent.

In this work, we propose a novel solution to address the problem of switching the robot's role automatically during a cooperative manipulation task. Additionally,

the robot does not need to know the final position of the object. This is practically desirable, since the motion trajectory of the object may require to be changed during task execution, depending upon the environment, obstacles or the physical limitations of the human and/or the robot.

The proposed work uses a prediction-evaluation method to estimate the confidence of prediction and use it to adjust the role of the robot. Our hypothesis stems from the observation that, in a human-human team performing a collaborative task, each human constantly predicts the other's motion. Based on how well the other person conforms to his predictions, the human can decide whether to lead him or follow him. We apply the same strategy to the humanoid robot. Another way of looking at this solution is, suppose if the robot is able to predict the human's motion accurately, it means that the robot has acquired an accurate model of the human's behavior. Hence, it can start behaving as a leader and proactively take the next action based on its prediction. However if the robot has not been able to predict the motion correctly, it is better for the robot to reactively comply with the human. This intuition sets the basis for adjusting the leader/follower role of the robot continuously and dynamically.

4.2 Experimental Platform

For the experiments we developed a platform which consists mainly of a Vicon motion capture system and a Nao humanoid robot. The table-lifting task consists of the human and humanoid robot lifting up a dummy table to a random height and keeping it down. Figure 4.1 shows the experimental setup. Only the positional information of the table is used for characterizing the task. We do not use force sensors because the table does not have a significant weight. The Vicon motion capture system provides precise position and motion information about the table. Motion of the robot hand is constrained to 1-D up-down motion. However, the proposed system can be easily extended to handle multiple dimensions. C++ is used at the front-

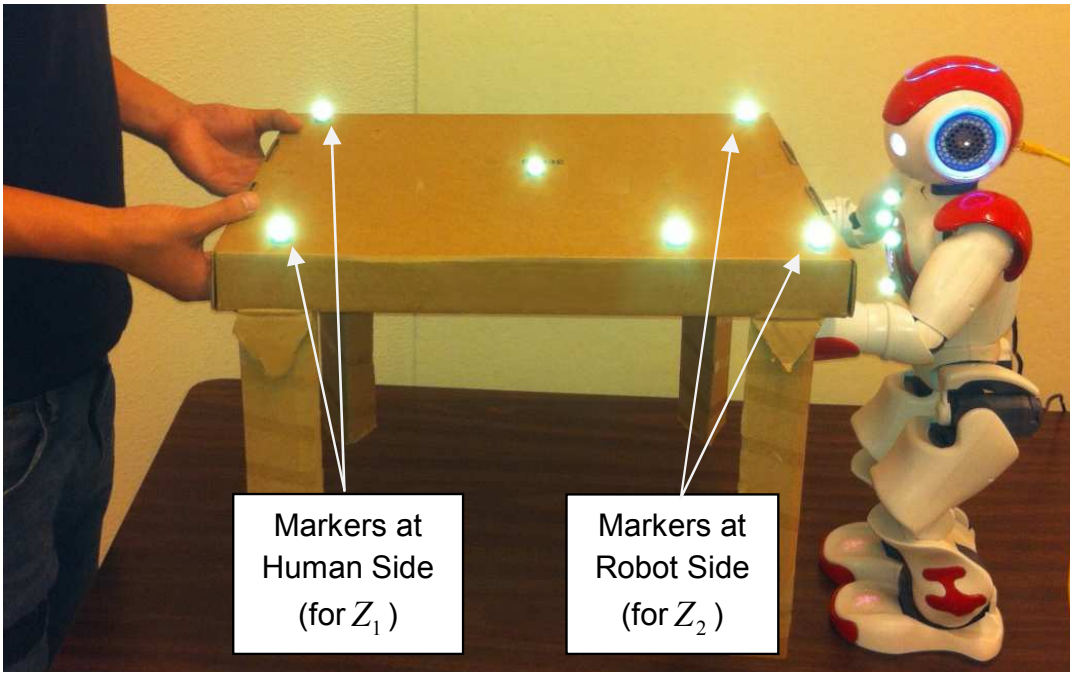


Figure 4.1: Experimental setup

end for communicating with the robot and MATLAB©is used at the back-end for processing data.

4.3 Methodology

Figure 4.2 shows the conceptual block diagram for the proposed system. The framework consists of the reactive controller, proactive controller and the behavior gain control blocks. As the name suggests, the reactive controller generates a reactive robot behavior based upon the current state of the environment. The proactive controller consists of a kalman filter (KF) based human motion predictor and an evaluation-based confidence generator. Based upon the observed human actions, the predictor estimates the position of the human in the next time-step, which decides the robot's proactive action. Additionally, it generates the confidence of prediction, which is the key in adjusting the role of the robot. Based upon the confidence value, the behavior gain control block mixes the reactive and proactive actions to generate

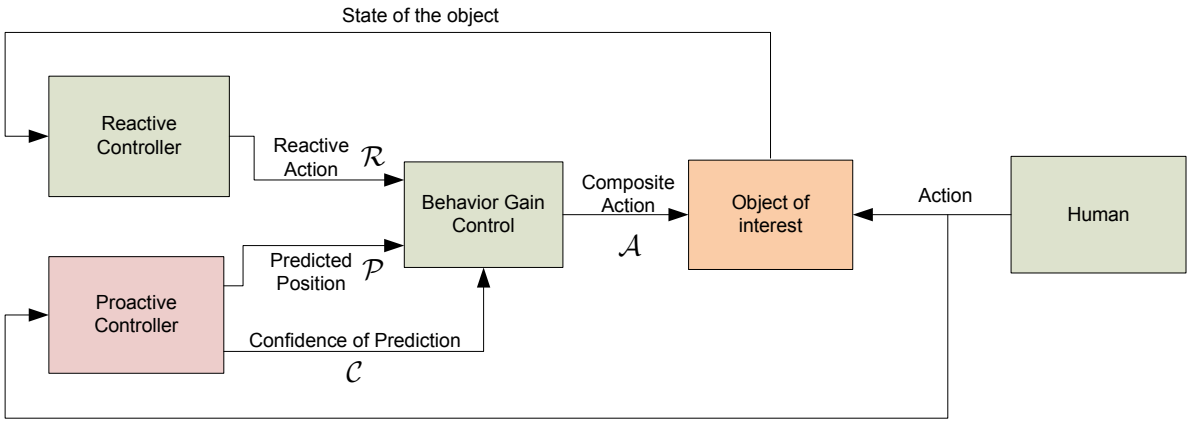


Figure 4.2: Proposed approach block diagram

a composite action which is taken by the robot. According to our hypothesis, the weight allotted by the gain control block to the proactive behavior varies directly as the confidence value. In the remainder of this section, we discuss the details of the proposed framework.

4.3.1 Reactive Controller

The reactive controller generates a reactive response by the robot to the observed state of the object. In the table-lifting task, this controller observes the position of the table and suggests a suitable action to perform so that a certain objective is achieved. For our experiments, the objective is to keep the table horizontal throughout the task. This can be accomplished using any generic feedback controller. However, we choose to use a controller learned from reinforcement learning for the following reasons :

- It is possible to learn a good controller in a short time.
- It compensates for the time needed to manually tune the parameters of a feedback controller.
- Objective of the task is very simple in the current experiment. However, in the future, we will consider complex tasks such as *keeping a bowl in the center of*

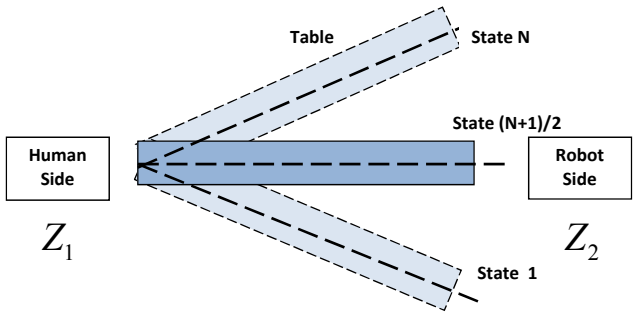


Figure 4.3: State representation for reinforcement learning

the table while performing the table lifting task. Complex tasks like these, have a long term reward to maintain for which reinforcement learning is most suited. Also, such high level objectives are much easier to specify using reinforcement learning.

In this work we use the discrete Q-learning algorithm. The Q-table update equation is given by

$$\Delta Q(s_t, a_t) = \alpha[r + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \tag{4.1}$$

where r is the reward, α is the learning rate and γ is the discount factor. For the task at hand, γ does not play a significant role, since there is no sense of a long term reward. The state of the environment is determined by the incline of the table at the given moment. This information is obtained from the motion capture system. Incline of the table is quantized into discrete number of states. Figure 4.3 shows a state space consisting of N states. The action space consists of a predetermined discrete set of commands which move the robot's hand-tip up or down by specified distances. The robot has to undergo an online learning phase to learn the Q-table. During this phase, it is assumed that the human remains comfortably stationary. To speed up the learning phase we use a simple guided reinforcement learning algorithm based on counting the number of state-action visits. Essentially, the action selection for

exploration is done on the basis of the number of visits to the particular state-action pair, instead of random action selection as in ϵ - greedy algorithms. The reinforcement learning algorithm is given below.

Algorithm 1 Guided Reinforcement Learning

1: Initialize $Visit(s_i, a_i) = 0 \ \forall i \in N$
2: Initialize Q-table $Q(s_i, a_i) = 0 \ \forall i \in N$
3: **while** Learning phase **do**
4: $t = timestep$
5: $s_t = getState()$
6: Select $a_t \leftarrow \text{argmin}(a)[Visit(s_t, a)]$
7: Take action a_t
8: $Visit(s_t, a_t) \leftarrow Visit(s_t, a_t) + 1$
9: $r = getReward()$
10: Update $Q(s_t, a_t)$ using (4.1)
11: **end while**

4.3.2 Proactive Controller

The proactive controller is the most important block of the proposed system. Role of the proactive controller is to keep a track of actions performed by the human and generate a prediction of the human's position in the next time-step, along-with a confidence measure for the prediction. For the prediction purpose, a Kalman filter is used. State of the KF x_k is given by

$$x_k = \begin{pmatrix} s_k \\ v_k \\ a_k \end{pmatrix} \tag{4.2}$$

The measurement model is given by

$$z_k = s_k + v \tag{4.3}$$

where s_k is the displacement of the human's end of the table (equivalently his hand-tip), v_k is his velocity, a_k is his acceleration and $v \sim N(0, R)$ is the measurement noise, all at the instant k. The measurement model can be rewritten as

$$z_k = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} x_k + v \tag{4.4}$$

We use the assumption that the acceleration of the human hand changes slowly throughout the motion since humans naturally try to minimize jerk. Note that this is not the same as using the minimum jerk model.

Hence, the state update equation can be written as

$$x_{k+1} = \begin{pmatrix} s_k + v_k t + \frac{1}{2} a_k t^2 \\ v_k + a_k t \\ a_k \end{pmatrix} + w \tag{4.5}$$

where $w \sim N(0, Q)$ is the process noise. For $t = 1$ the system model can be rewritten as

$$x_{k+1} = \begin{pmatrix} 1 & 1 & 0.5 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} x_k + w \tag{4.6}$$

Based on the state estimate $\hat{x}_k$, the human's position at the next time-step can be predicted as

$$\hat{s}_{k+1} = \hat{s}_k + \hat{v}_k t + \frac{1}{2} \hat{a}_k t^2 \tag{4.7}$$

The variance of the measurement noise (R) is initialized to 0.7 which corresponds to the uncertainty in measurement obtained by the Vicon system. Using this KF, it is possible to get nearly accurate predictions of the human's motion.

For obtaining the confidence of prediction, we derive inspiration from [39], wherein the authors proposed a technique, to obtain a confidence measure based on the statistical properties of the residuals between the predicted measurements and the observed measurements. In our technique, the KF provides a state estimate and an associated covariance matrix. Firstly, we marginalize the covariance matrix to include only the 1-D variance associated with the position prediction, say ρ. Let the predicted position be $\hat{s}_k$. Then, we evaluate the likelihood of the observed measurement z_k using an unnormalized Gaussian distribution given by

$$\mathcal{L}_k = \exp\left(-\frac{(z_k - \hat{s}_k)^2}{2\rho^2}\right) \tag{4.8}$$

We choose an unnormalized Gaussian distribution to make $0 < \mathcal{L} \leq 1$. It can be seen that $\mathcal{L}$ would give us a direct measure of confidence about the prediction based on the evaluation of the previous prediction against the true measurement. However, considering only the last step measurement error is not sufficient. For the confidence measure, we introduce a function given by

$$\mathcal{C}_{k+1} = \frac{\mathcal{L}_k + \phi\mathcal{L}_{k-1} + \cdots + \phi^{k-1}\mathcal{L}_1}{1 + \phi + \cdots + \phi^{k-1}} \tag{4.9}$$

The subscripts denote the time-steps at which they were obtained. Hence, $\mathcal{C}_{k+1}$ is the confidence of prediction for the next time-step, that considers all the likelihoods observed previously, weighted by the forgetting factor ϕ, where $0 < \phi \leq 1$. This function can be implemented recursively. Also, it can be seen that the denominator is for normalization.

4.3.3 Behavior Gain Control

At a given time step k, let the reactive controller suggest a next-step action $\mathcal{R}_{k+1}$ and the proactive controller suggest a next-step action $\mathcal{P}_{k+1}$. Let the confidence of this prediction be $\mathcal{C}_{k+1}$. The gain control block combines these together to form a

composite action $\mathcal{A}_{k+1}$ given by

$$\mathcal{A}_{k+1} = \mathcal{C}_{k+1}\mathcal{P}_{k+1} + (1 - \mathcal{C}_{k+1})\mathcal{R}_{k+1} \tag{4.10}$$

This action is taken by the robot at time-step $k+1$. The inspiration for this form has been taken from [32]. Note that because $0 < \mathcal{C} \leq 1$, the robot does not act as a pure leader or pure follower, but has characteristics of both in different amounts.

If the confidence of prediction $\mathcal{C}_{k+1}$ is high, larger weight is allotted to the proactive action. Hence, the robot's action has leader-like characteristics. If the robot is not very confident about the prediction, larger weight is allotted to the reactive behavior and the robot's action seems follower-like. Since the system works in real time, the change of behavior is dynamic and automatic.

4.4 Experimental Results

In this section we present the experiments performed and the results obtained.

4.4.1 Learning the Reactive Controller based on Q - Learning

For Q-learning, a state-action space consisting of 5 states and 5 actions was arbitrarily chosen. The reward r was decided as

$$r = (|Z_2 - Z_1|)_k - (|Z_2 - Z_1)|)_{k+1} \tag{4.11}$$

where Z_1 and Z_2 represent the position of the human-end and the robot-end of the table respectively.

Hence, if the slant of the table is decreased, the robot receives a positive reward. The action set consists of actions {+2,+1,0,-1,-2}, which correspond to the direction and magnitude of the robot's motion by a defined position step. The position step was set to be 2 cm, since it is the smallest precise movement that can be performed by

the robot's arm. Values of the reinforcement learning parameter used were, learning rate $\alpha = 0.9$ and discount factor $\gamma = 0.2$.

Ten trials were performed to test how quickly the algorithm could converge to an optimal policy. Median value for number of iterations to converge was 36. The longest episode took 62 iterations before it could converge. Hence, the learning could converge approximately within 40 trials. Each learning trial took about 5 minutes to complete.

4.4.2 Prediction

The previously described KF is used for predicting the human motion one time-step ahead. Each time step is typically about 100 ms, which is the minimum time required for the robot's arm to move from one position to another. Figure 4.4 shows the predicted and observed values of position, velocity and acceleration.

The predicted position is calculated from (4.7). True velocity and accelerations calculated from the actually observed positions, and are shown in the figure for comparison with the predicted velocities and accelerations.

It can also be observed from Figure 4.4 that the predictions are inaccurate during the initial steps of the motion. After about 10 time steps the estimates improve. It can be seen that the difference between the predicted velocity and the calculated velocity is very small. The calculated acceleration nearly remains centered at 0 with small changes. This partly justifies our assumption that the acceleration remains nearly constant.

4.4.3 Confidence Measure

Figure 4.5 shows how the confidence ($\mathcal{C}$) of the prediction varies throughout the task, along with the position predictions and observations.

It can be seen that initially, when the task has not begun and the table is still, the

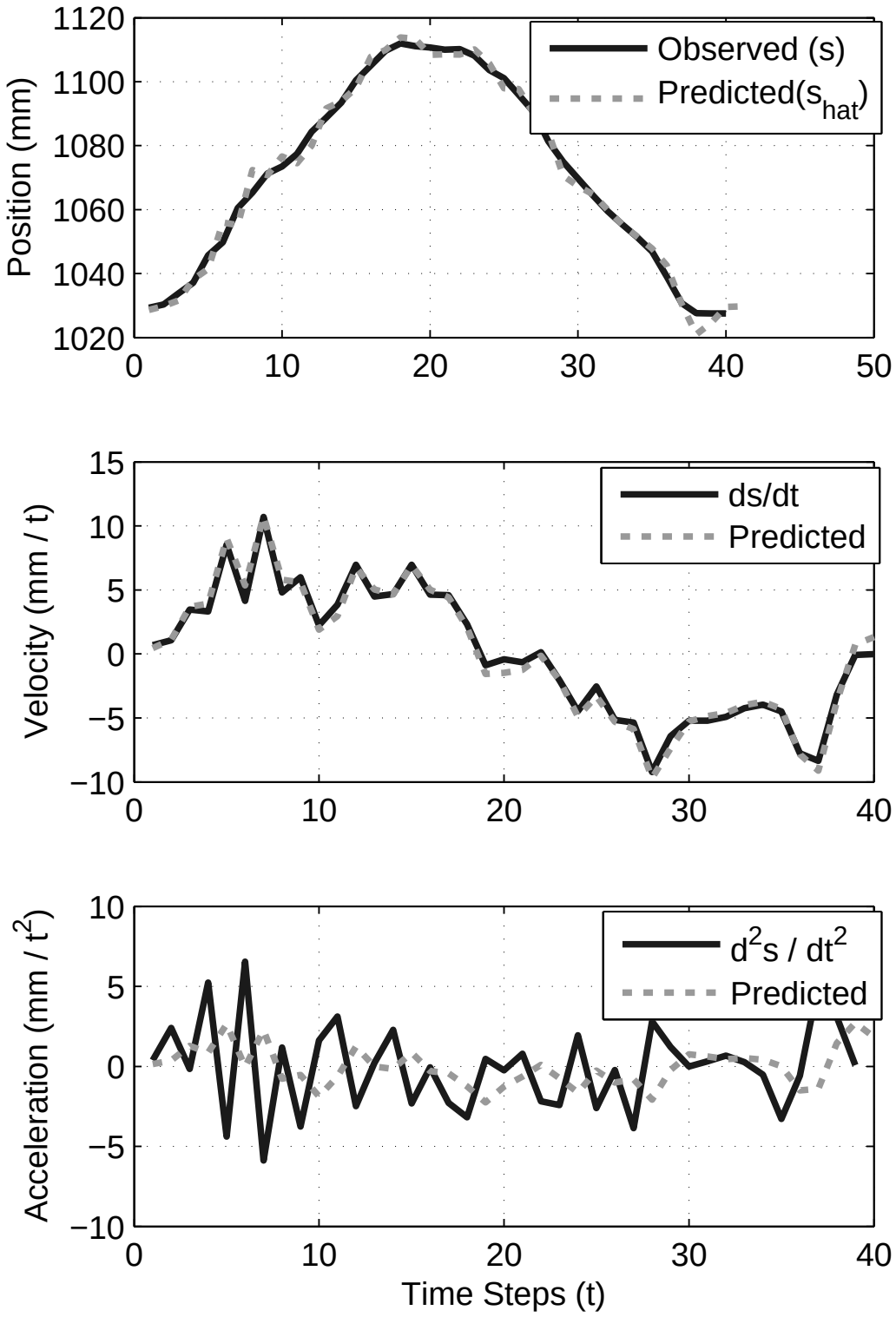


Figure 4.4: Predictions obtained from KF

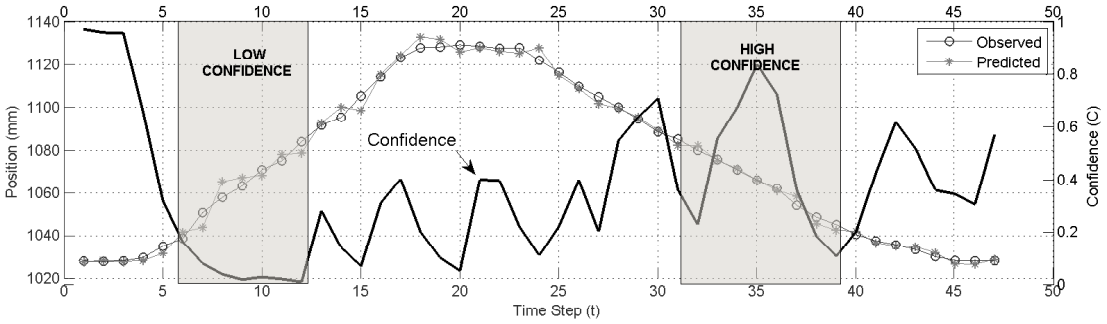


Figure 4.5: Confidence value with predictions

predictor accurately estimates the motion to be zero which causes the high confidence value at the beginning. Once the trial starts, in the initial steps, the predictions are inaccurate because of the drastic change in the motion model. This causes confidence value to drop down suddenly. The reactive controller of the robot becomes dominant in this region. As the predictor gains knowledge about the motion, the predictions go on improving. As the predictions improve, the confidence values also improves. As a result the proactive behavior becomes more dominant.

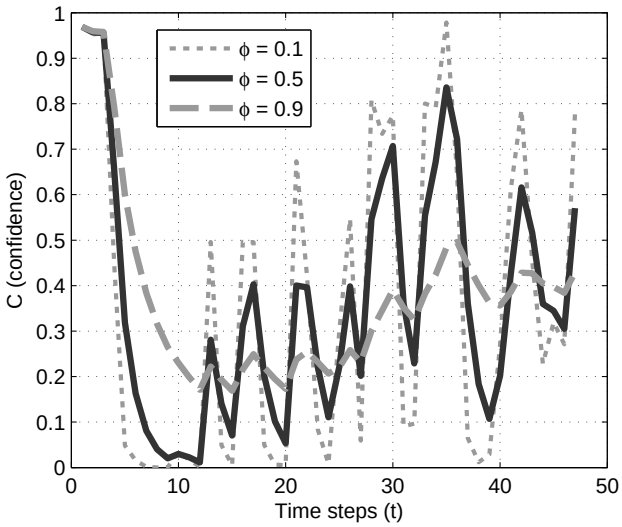


Figure 4.6: Effect of the forgetting factor on confidence

Figure 4.6 shows the role of the forgetting factor ϕ in determining the confidence. Since it is not possible to reproduce the exact same trajectory during the task, the

confidence trajectories shown in fig. 4.6 are computed offline step-by-step using data collected from a table lifting task. As seen from (4.9), a low value of ϕ means that the predictor allots a small weight to older likelihood estimates. Thus, $\mathcal{C}$ mostly depends upon the recently observed $\mathcal{L}$. Hence, if the likelihood values $\mathcal{L}$ change quickly, it causes $\mathcal{C}$ fluctuate heavily. Using a similar reasoning, a large value of ϕ causes the confidence measure to settle very slowly. Hence the robot cannot adapt to the motion changes quickly and generates high confidence values even for incorrect predictions. A good value for ϕ which gives a good tradeoff between smooth variation and adaptability for $\mathcal{C}$ was found to be 0.45.

4.4.4 Handling Irregular Cases

One of the major improvements our system offers over most existing systems, is that, no assumption has been made regarding the trajectory of the entire motion. The human has the right to change the trajectory at any point of time, during the trial. Figure 4.7 shows a case where the motion of the human is not typical. Instead of lifting up the table and keeping it down continuously, the human chooses to take a pause while lifting the table up. Because of this, an abrupt change of motion can be seen around time-step 15. The confidence value drops to zero in 3-4 time steps. During this phase, the robot starts behaving as the follower and simply tries to make the table horizontal using the reactive controller. As the human continues to keep still, the predictor learns this model and predicts zero movement. Hence, although the confidence is high and the robot is the leader, there is no proactive action since the predicted change in position is zero. Again at time-step 35, the human starts moving the table upwards. Again, the robots switches from leader to follower based on the confidence value. Once the motion has been stabilized the robot maintains a confidence value centered somewhere around 0.5.

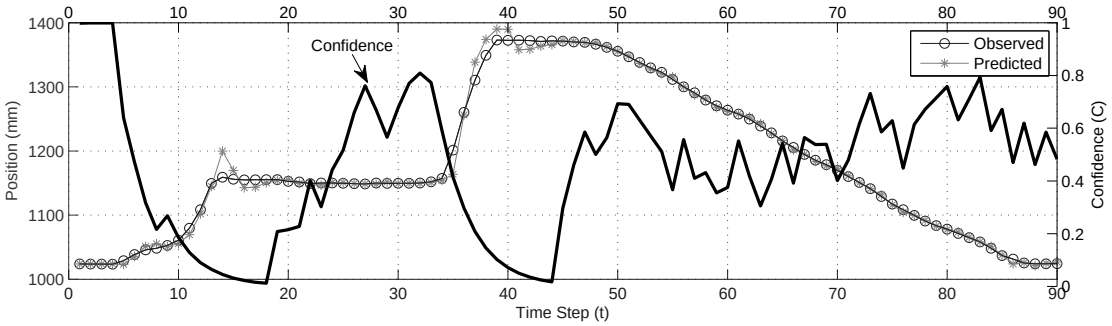


Figure 4.7: Irregular case

4.4.5 Overall System Performance

Table 4.1: Average RMSE

Subject	Avg. RMSE w/o Prediction (mm)	Avg. RMSE with Prediction (mm)
1	19.139	12.967
2	23.567	16.591
3	24.872	18.418
4	20.085	15.391
5	22.432	17.684

In this experiment we evaluate the improvement offered by our system for the table lifting task. If Z_{1t} is the position of human side of the table and Z_{2t} is the position of robot side at any instant t, then the objective is to minimize the absolute error given by

$$AbsoluteError = \sum_{t} |Z_{1t} - Z_{2t}| \tag{4.12}$$

We use the motion capture system to record the trajectories of the human and robot table ends. Figure 4.8 shows these trajectories for cases where the the proposed system was used (case I : with predictions) and the case where only the reactive

controller was used (case II : without predictions) The figure also shows the absolute error calculated for the two cases. We use the root mean square error (RMSE) to characterize the performance.

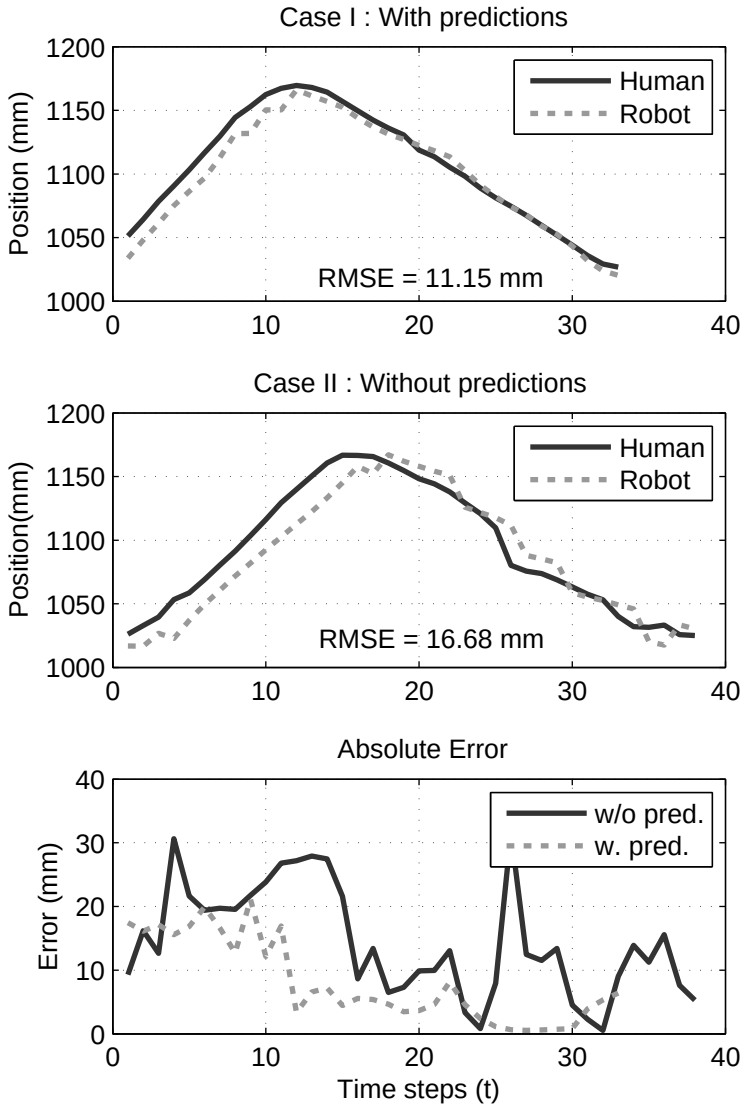


Figure 4.8: Comparison

The following observations can be made from Figure 4.8

- The RMSE for the case I is less than RMSE for case II.

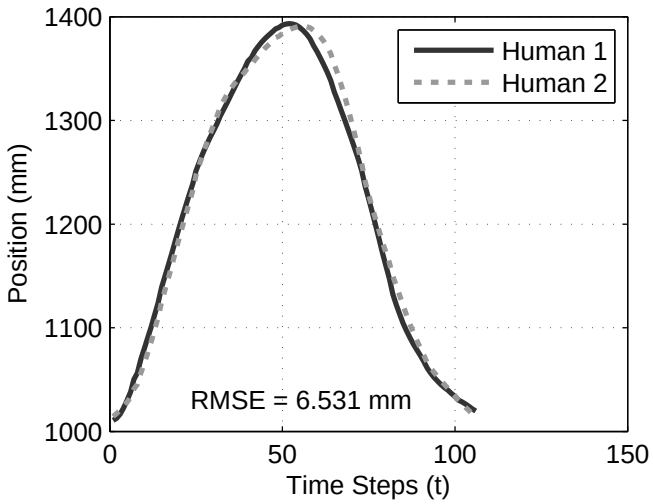


Figure 4.9: Human-human team lifting the table

- The motion observed for case I is smoother than that of case II.
- The absolute error is lower in case I.

Quantitative results are provided in table 4.1 for multiple users. 5 human subjects were asked to participate in the table lifting task with the robot, one at a time. Each person was asked to lift up the table to a random height and keep it down for 10 trials. Totally, for both the cases, 100 trials were acquired. The table shows the average RMSE for the 10 trials observed for each subject, for each case. It can be seen that, for all the users, RMSE is lower when the proposed approach is used as opposed to a simple reactive approach. Hence a definite improvement can be observed.

4.5 Discussions

Figure 4.9 shows the performance of two humans performing the table lifting task. For the sake of comparison with the human-robot team, the RMSE observed was 6.531 mm.

The motion of the robot is jerky when its reactive behavior is dominant, because of the fixed step sizes. The design of our system is such that the prediction accuracy

influences the confidence of prediction. Because of this, many interesting possibilities follow. Better predictions result in better confidence values which allow for proactive robot behavior. Hence, if the human keeps moving smoothly as the robot expects him to move, the motion of the robot is also smooth. This in-turn causes smoother motion of the table as a whole and hence smoother motion of the human, thus resulting in better predictions. However, if the motion of the human is jerky, then the robot is unable to estimate the motion accurately, and hence does not allow for leader behavior. The predictions are not fully utilized in such cases and reflects poor performance. Thus, the results are not only influenced by the robot's performance alone, but also by the human performance. Especially, subject 1 had been working with the system for a longer time than others. Hence, the results for subject 1 were better compared to other human subjects.

In Figure 4.8 we could observe in case I, the trajectory is much smoother when the human is placing the table down as compared to moving upwards. This is because, inherently, the robots motion while lifting the table against gravity is jerky because of the internal control characteristics. This induces some jerks in the human motion also since they are coupled by the table. Because of this, the prediction suffers, which causes lower confidence levels. But while moving downwards, the robot is able to move very smoothly which allows the human to move smoothly and hence the system is utilized to its full potential resulting in better performance. It can also be speculated that sophisticated velocity or torque controlled robots would yield smoother motions and offer better improvements in performance using the proposed technique.

Due to the limitation in the control speed of robot, we could obtain atmost 10 motion capture samples per second. With a faster robot, more samples could be obtained per second which would improve the quality of predictions.

Finally, our work can also be easily extended to proactive teleoperation. The teleoperated robot can choose to take a proactive action based on the confidence

values which could reduce the effect of time delays observed in teleoperation and increase transparency.

Thus, we have seen how implicit communication by human motion is possible, and can be used by robots to determine their role in a collaborative task. We have also seen how robots can utilize predictions to take a leader-like role in collaborative tasks. In the next chapter, we shall see how robots can infer human intentions based on his actions and also predict human's motion for pre-emptive task planning.

CHAPTER 5

INFERRING HUMAN INTENT FOR PREDICTIVE TASK PLANNING

As we have seen in the last chapter, human-robot collaborative tasks are the best examples where implicit communication is involved. In the table lifting task, we do not make any assumptions about pre-existing robot knowledge. If we assume that the robot has a knowledge of task primitives (such as knowing *how to pickup a bowl*), humans and robots can perform collaborative tasks which require a higher level of representation and reasoning. An example of such collaborative task could be a human and a robot collaboratively setting up a table for dinner. This task can be decomposed into primitives such as *pick-up bowl*, *place bowl*, *pick-up spoon* and so on.

In the final part of this thesis, we investigate implicit communication by human motion, in a high level human-robot collaborative task. Specifically, we investigate how the robot can utilize long term human motion predictions for predictive task planning.

5.1 Related Work

For achieving tasks which can be decomposed into robot primitives, task planners are generally used. Task planning involves mapping a given set of high level instructions to a sequence of robotic tasks [40]. Task planning is extensively used in industrial settings. The planner generates plans based on the required goal and the current observation of the robot's environmental state. Such planners can be employed only

in highly structured and highly deterministic domains.

Instead of waiting for the environmental state change to actually occur, it would be advantageous if planning takes place based on prediction of the state change, given some observations. Some of the earliest works in this area was investigated by Dean *et al* [41]. The intuition that humans use predictions for planning is supported by studies in neuro-science, specifically [42] in which the authors report different areas of the brain being used in planning for predictable and non-predictable events. It is financially risky to use predictive planners in the industry, since wrong predictions can lead to wrong plans, and hence there is limited research in this area. However, in human-robot collaborative tasks, purely reactive planners can considerably slow down the task performance. This is because the robot would have to wait untill the human has somehow influenced the environmental state, often by completing a part of the task. It is possible for the robot to predict outcome of human actions and thus infer his intent. Based on such predictions, the robot could generate plans and start working on them immediately.

The planners that consider human presence in the robot's environment are called human aware task planners. The simplest example of human aware task planning is human aware navigation. Works such as [43] provide a framework for mobile robots to plan their path from one place to another, while predicting and avoiding the human's path. Human motion prediction is also important for the sake of the human's safety. In the work presented by Kulic *et al.* [44], a danger criterion is minimized by the robot in the planning stage to avoid collisions with a human in an object hand off task. A comprehensive human aware task planning framework is proposed by Alami *et al.* [45] in which the authors present a three layered HRI architecture that takes care of various spatial, temporal and even social constraints while planning a task.

In this work, we setup a restaurant kitchen-like scenario where a human and a robot work collaboratively to assemble various *ingredients* to prepare a *dish*. Sakita

et al. have presented a work in which human's gaze direction is used by the robot to infer his intent, and plan a cooperative assembly task [46]. In our work, we propose using the direction of the human's hand motion to infer his intent and plan the task predictively. Our framework derives inspiration from the intuition that, we humans begin to plan our tasks based on our prediction of what the other person might do next, by observing his movements.

5.2 Experimental Setup

The restaurant-kitchen task basically involves assembling a set of ingredients in a bowl, given the menu, recipes and customer order. Both human and the robot know the menu, recipes, and the customer's order. The human and robot are in charge of different ingredients. We assume that there will be 3 customer orders at a time. Hence, the human-robot pair will be working on 3 bowls at a time. The only thing that the robot does not know is, which bowl should hold which dish. The human is in charge of deciding this. Hence, once the robot figures out which bowl corresponds to which dish, it can take an appropriate action towards that bowl.

However, inferring the dish based upon the ingredients observed in the bowl, is a simple problem. In this work, we propose that the inference should occur based on the prediction of human's hand-motion. Predictive task planning is achieved by utilizing the prediction on which bowl the human is reaching towards. Thus, even before the human can put an ingredient in the bowl, the robot can already infer what dish is the particular bowl going to be for, and can start acting immediately. Apart from the bowl on which prediction is made, the robot can also infer the possible dishes planned in the other bowls, and can decide its actions towards them. Figure 5.1 shows the complete setup. The human hands, the ingredients and the bowls are tracked by the Vicon motion capture system.

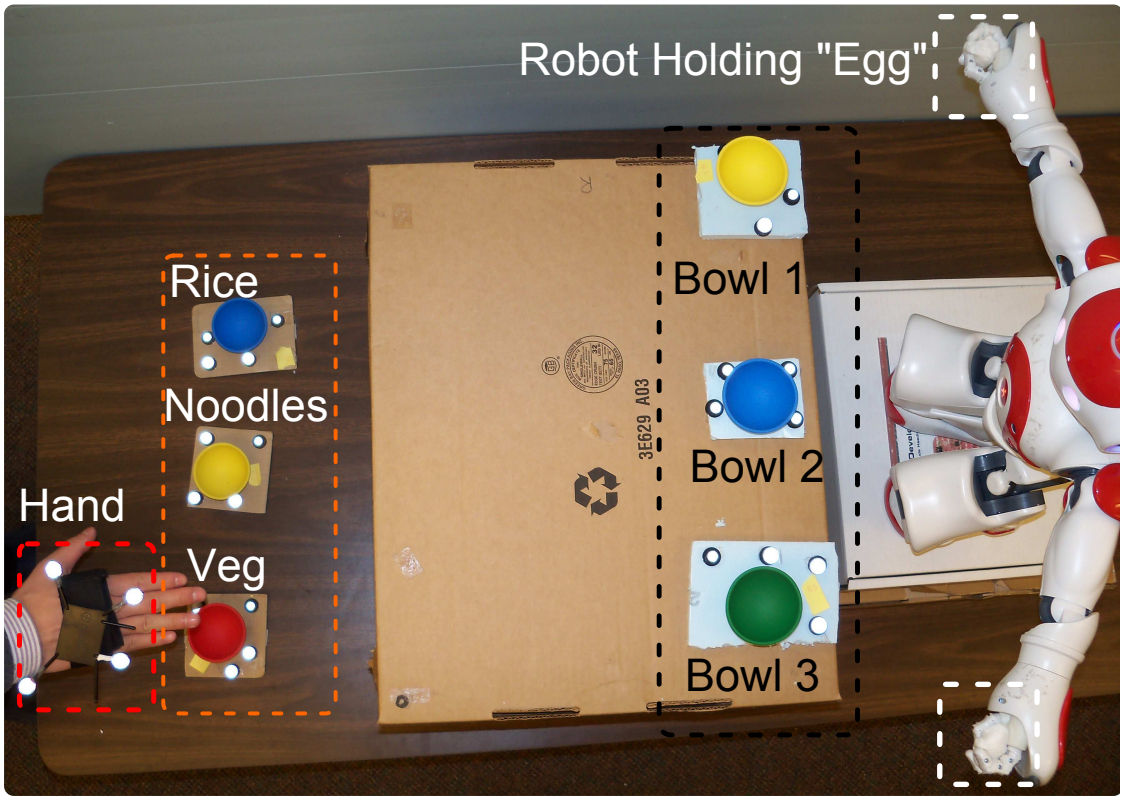


Figure 5.1: Experimental setup

5.3 Methodology

Figure 5.2 shows the block diagram for the proposed framework. The remainder of this section explains the block diagram in detail.

5.3.1 Motion Capture Program

The motion capture program obtains the position and motion information of the bowls and the human hands from the motion capture system. If the human wants to *pick-up* an ingredient, he has to put his hand over that particular ingredient and hold for 1 second. He is notified by a beep upon success. If he wants to *drop* that ingredient in a particular bowl, he has to hold his hand (containing the ingredient) over the bowl for 1 second. He is notified by a beep on success and the particular ingredient gets added to that bowl. *Bowl contents* is updated accordingly.

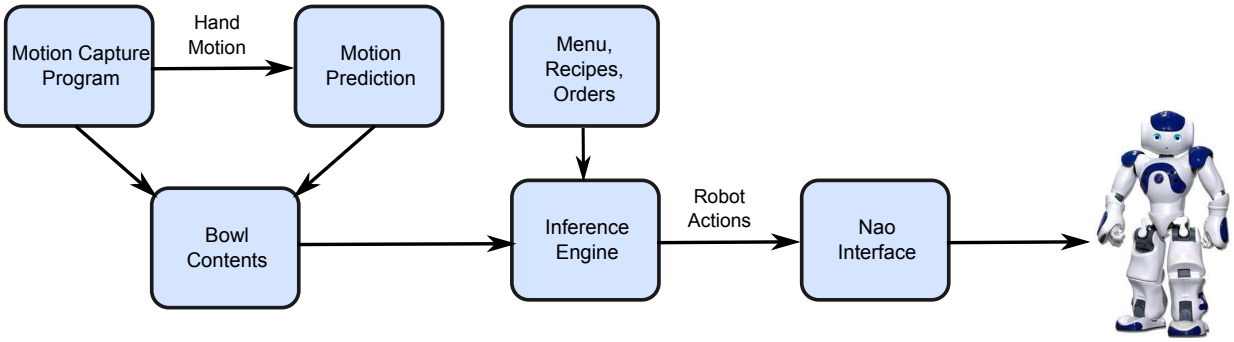


Figure 5.2: Block Diagram

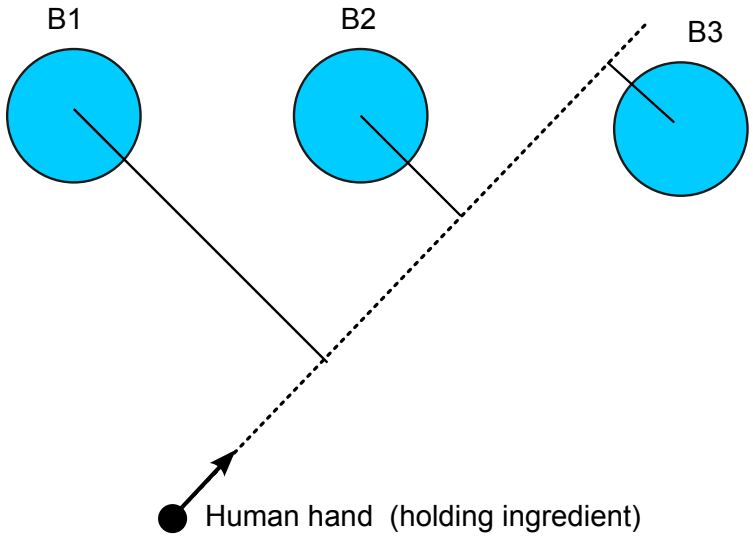


Figure 5.3: Hand Motion Prediction

5.3.2 Motion Prediction

The motion prediction program tracks the human hand only after he has "picked-up" an ingredient from the bowl. The predictor basically fits a line through the tracked hand positions. When the distance of one of the bowls from this line falls below a threshold, the system predicts that the human would put the ingredient into that particular bowl. Only the last 3 positions of the human hand are used for line-fitting, since older data points are not relevant for making predictions. Also the predictions are made only when *significant* hand motion is observed. Fig. 5.3

illustrates the motion prediction strategy. The solid black line represents the observed trajectory of the human hand. The dotted line represents the line fitted through the last 3 observations. When the perpendicular distance of any bowl falls below a predetermined distance threshold (d_{thresh}), prediction is made for that particular bowl and the *bowl status* is updated. Additionally a directional constraint is added to ensure that the human hand is moving towards the bowls and not away from them.

5.3.3 Inference Engine

The inference engine generates plans and subsequently robot actions based on *bowl status*. Based on the ordered dishes, all valid plans for each bowl are generated. Depending upon the *bowl status*, which is updated by the motion capture and prediction programs, the invalid plans are ruled out. The planner then searches for the *imperative* robot actions in the remaining set of plans for each bowl. Once an imperative action is found, the engine triggers robot actions available in the *Nao robot Interface*. An example of the inference and planning process is given in the next section.

5.4 Results

The menu consists of three dishes $\{FR, FN, SP\}$ (abbreviations for fried rice, fried noodles and soup). The ingredients available are $\{rice, noodles, veg, egg\}$ (abbreviations to rice, noodles, vegetables and eggs). The human is in charge of handling rice, noodles and vegetables. The robot is in charge of handling eggs.

The recipes are given below :

$FR = \{rice, veg, eggs\}$; $FN = \{noodles, veg\}$; $SP = \{noodles, eggs\}$;

5.4.1 Inference

Suppose the customer orders all the three dishes that is FR, FN and SP. Fig. 5.4 shows the possible plans generated by the planner for this particular order.

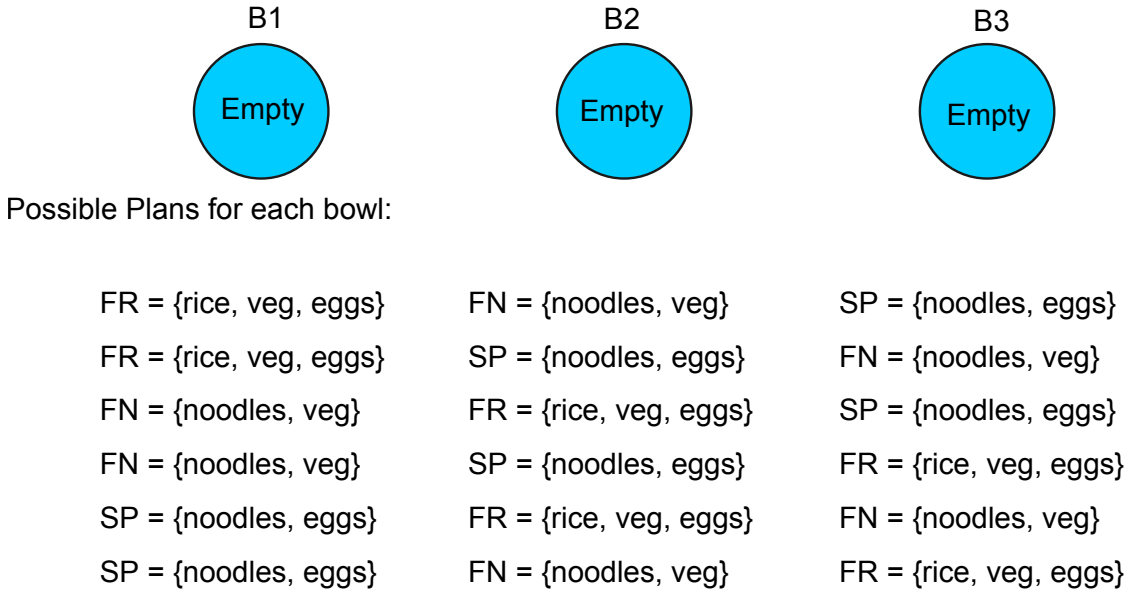


Figure 5.4: Planning - all bowls are empty

Initially all bowls are empty, so all possible plans are valid. Suppose, the human decides to assemble *FR* in bowl *B2*. He picks up "rice" and starts moving his hand towards bowl *B2*. The prediction program starts tracking the human hand and makes the prediction that he is moving towards bowl *B2*. *Bowl status* is updated, with bowl *B2* now containing "rice". Based on the updated bowl status, the inference engine rules out the invalid plans and generates a new set of plans, as shown in Fig. 5.5. Now, imperative actions are searched for in the new set of valid plans. For example, in Fig. 5.5, we can see that "eggs" are present in all plans for bowl *B2*. The inference engine then sends a command to the robot to put eggs in bowl *B2*.

5.4.2 Motion Prediction

As we have discussed, the framework relies on motion prediction for predictive task planning. Humans typically take an action based on their prediction, only if they have predicted it with a *good confidence*. The distance of a bowl from the fitted line acts as the confidence measure for prediction. In this work, a threshold (d_{thresh}) determines how confident the prediction should be, before the action can be taken.

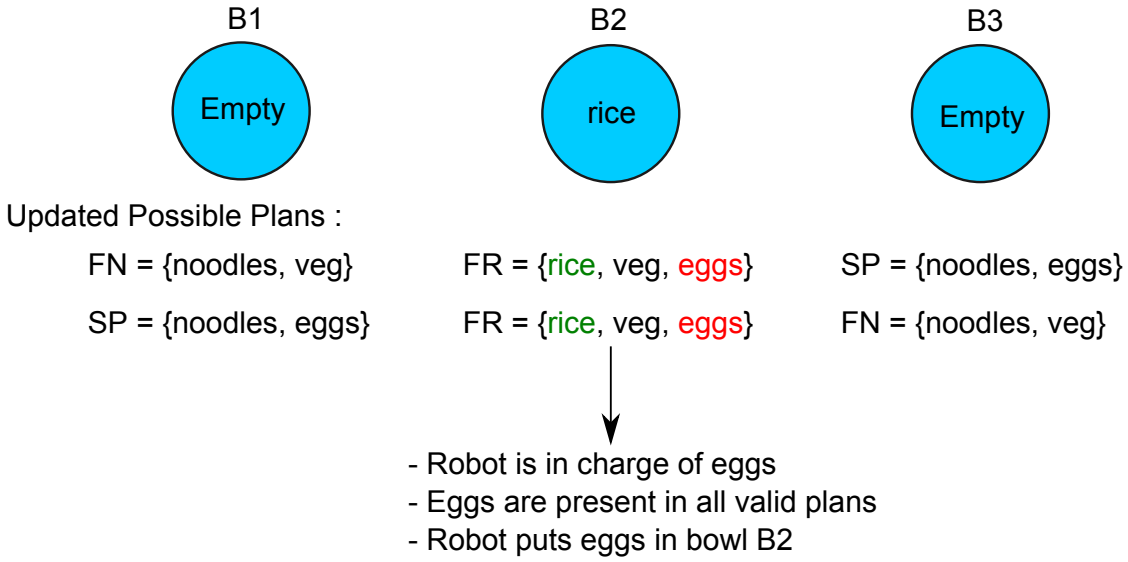


Figure 5.5: Planning - bowl 2 contains "rice"

If the threshold is never reached, then the robot simply waits for the human to actually put the object in the bowl. We perform experiments to evaluate the accuracy and usefulness of motion prediction for two different arrangements of the bowls with respect to the ingredients. Fig. 5.6 shows the position of the ingredient with respect to the bowls for the two setups. The orange circle indicates the ingredient, whereas the blue circles indicate the bowls. The distance between two bowls is more than 250 mm. In the first setup, we can see that the direction of each of the bowls is quite different with respect to the ingredient. In the second case, the directional separation of the bowls is less than the previous case.

The most important factor that determines the prediction results is the threshold (d_{thresh}). The threshold value basically specifies how close the bowl should be to the predicted human hand path/line, before a prediction can be made. We add a condition that if two bowls are within the threshold simultaneously, then a prediction is not made (single-bowl condition). The threshold value is varied from 10 to 200 in steps of 50. Two subjects are employed for the experiments. For every setup, for a given threshold value, the subject has to pickup the ingredient and drop it in

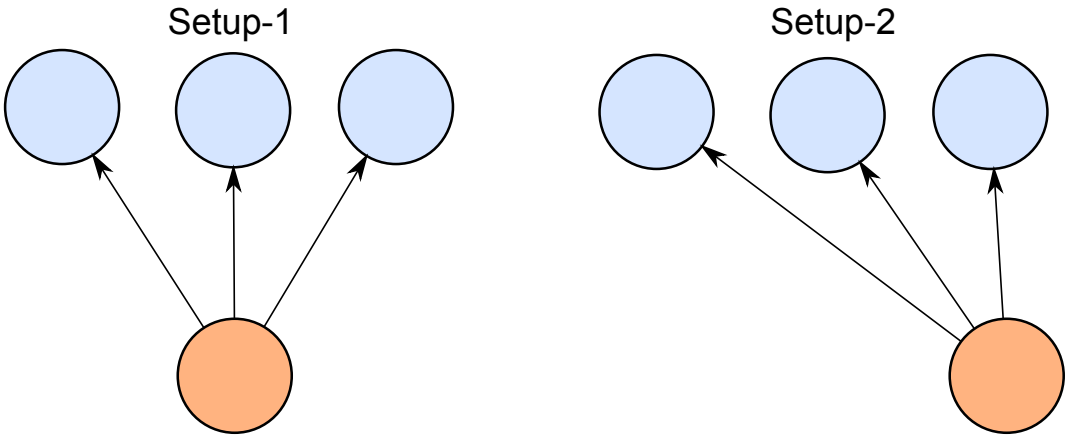


Figure 5.6: The two different experimental setups

the bowls for 15 trials. For each trial, the system may give a correct prediction, an incorrect prediction or no prediction at all. Let each trial start at time $t = 0$. We define $t_{predict}$ as the time when the prediction is made. We also define t_{finish} as the time when the trial ends i.e. the subject drops the ingredient in the bowl. We calculate lead time as

$$Lead\ Time = t_{finish} - t_{predict} \tag{5.1}$$

We calculate the percentage lead as

$$Lead(\%) = \frac{t_{finish} - t_{predict}}{t_{finish}} \tag{5.2}$$

Table 5.1 shows the prediction results (correct predictions (%), no predictions(%), wrong predictions (%), lead time(s) and lead (%)) for different values of threshold for setup-1. Results are given separately for each of the two subjects. Table 5.2 shows the prediction results for setup-2.

Observations

From Table 5.1, we can see that when the threshold value is very small, predictions are not made most of the times. For calculations of lead time and percent lead, incorrect and no prediction cases are excluded. We can see that for small values of

Table 5.1: Setup 1

	Subject 1					Subject 2				
Threshold d_{thresh} (mm)	Correct Pred. (%)	No Pred. (%)	Wrong Pred. (%)	Lead Time (s)	Lead (%)	Correct Pred. (%)	No Pred. (%)	Wrong Pred. (%)	Lead Time (s)	Lead (%)
10	33.34	66.67	0	0.51	22	66.67	33.34	0	0.32	15.49
50	100	0	0	0.81	33.92	93.34	6.67	0	0.97	36.41
100	93.34	0	6.67	1.27	43.53	100	0	0	1.13	42.23
150	100	0	0	1.33	45.1	93.34	0	6.67	1.01	38.14
200	66.67	33.34	0	0.74	32.69	66.67	33.34	0	0.56	31.89

Table 5.2: Setup 2

	Subject 1					Subject 2				
Threshold d_{thresh} (mm)	Correct Pred. (%)	No Pred. (%)	Wrong Pred. (%)	Lead Time (s)	Lead (%)	Correct Pred. (%)	No Pred. (%)	Wrong Pred. (%)	Lead Time (s)	Lead (%)
10	46.7	53.34	0	0.53	21	60	33.34	6.67	0.12	8.24
50	93.34	6.67	0	1.14	40	60	26.67	13.34	0.49	15.71
100	86.67	0	13.34	1.37	46.48	66.67	13.34	20	0.81	38.23
150	93.34	6.67	0	1.06	43.32	86.67	13.34	0	0.98	36.19
200	46.66	53.34	0	0.69	31.02	46.66	53.34	0	0.56	21.89

thresholds, the prediction is made quite late. Hence lead times and percentages are small. Lead times and percentages improve as d_{thresh} is increased. Since the threshold is larger, the system can make predictions at an earlier time. For moderate values of thresholds (100-150 mm), best results for lead is observed. However, there is a tradeoff with incorrect predictions. For large values of threshold (200 mm), the number of no predictions increase since the threshold value may be crossed simultaneously by

two bowls leading to no predictions. Even if predictions are made, they are made late, only when the single bowl condition is reached. For setup 2, since the directions of bowls with respect to the ingredient are very similiar, the occurrences of wrong predictions and incorrect predictions are increased. However, the trend is similar to setup 1.

From the results, we can say that the motion prediction algorithm can work quite reliably since the number of wrong predictions is very low (less than 10%). No predictions are better than wrong predictions, since wrong predictions can lead to incorrect robot actions, thus disrupting the task. Ofcourse, the threshold value needs to be adjusted for different setups. For this particular setup a threshold value of about 100-150 mm should be sufficient.

Lead

We see that the lead time given by the system is about 1 second, for tasks of length 2-2.5, which is about 40% lead time on an average for a well tuned system. For tasks where the human is slow and the robot is fast, such an improvement definitely improves task performance. Figures 5.7 and 5.8 show two cases where the robot takes action based on human motion prediction. We can see that the robot starts taking action when the human hand approximately completes half of the motion path.

5.5 Discussions

In this chapter we have seen how robots can infer human intent by using motion predictions. As compared to the table-lifting task, this task necessitated longer term predictions by the robot. We used a simple technique for human motion prediction based on line-fitting and thresholding. The threshold value plays an important role in determining how confident the robot needs to be before it can take the action. Experiments showed the influence of threshold value on the prediction results.

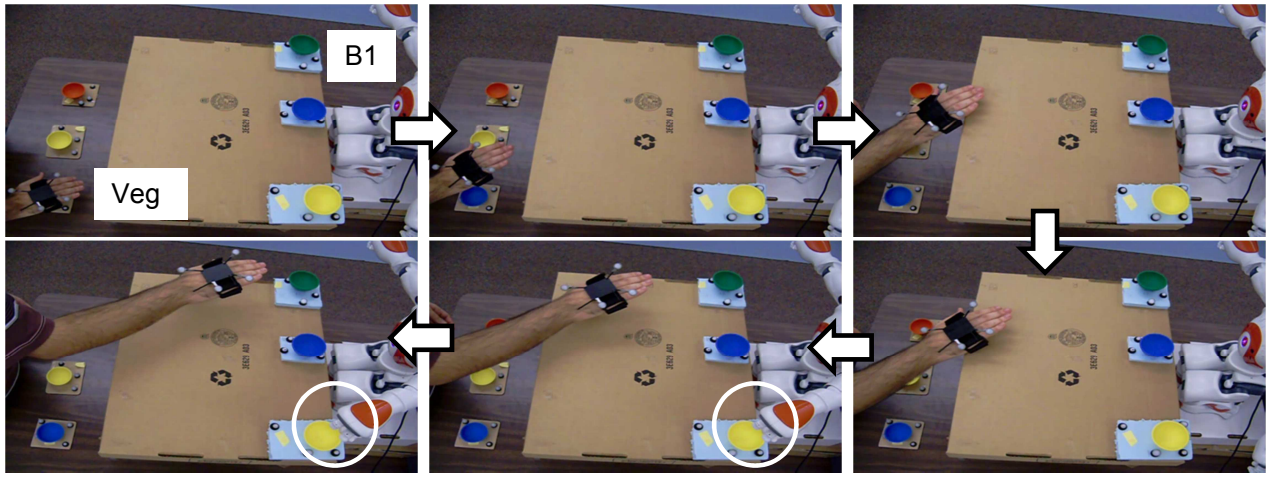


Figure 5.7: Human putting "veg" in bowl B1. The circles in the last 2 frames highlight the robot taking action

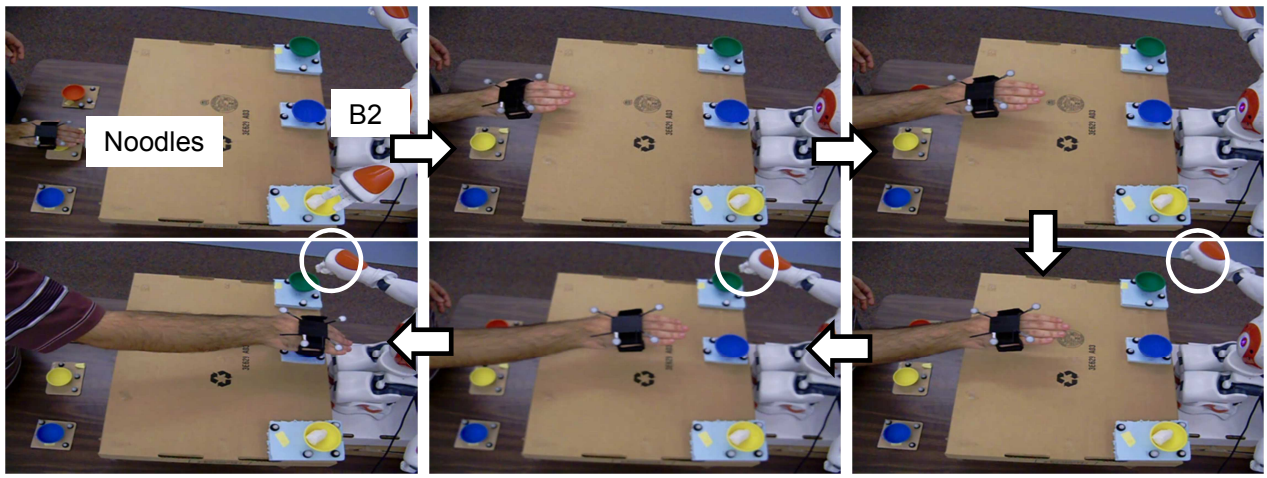


Figure 5.8: Human putting "noodles" in bowl B2. The circles in the last 3 frames highlight the robot taking action

CHAPTER 6

CONCLUSIONS

Human-robot interaction is possibly the key factor in making personal service robots a reality. For facilitating natural HRI, humans and robots need to communicate effortlessly. In this work, we investigated the role of human motion in both explicit and implicit communication.

As an example of explicit communication, we considered the learning from demonstrations problem in both joint angle space and the task space. For the joint angle space learning, we developed a platform for implementing and evaluating a learning by imitation framework for the humanoid robot to learn hand gestures. We collect the joint angle trajectories of the human arm motion. The imitation learning framework makes use of DTW to temporally align multiple trajectories. Weighted averaging is applied to these signals to get a generalized version of the joint angle trajectory for a particular hand gesture. We evaluated the imitation learning framework by attaching markers to the humanoid robot and compared it to the human motion. It was found visually and experimentally that the algorithms perform well for various hand gestures. As an extension to this work, we considered how to handle the missing data problem while learning arm gestures. We proposed and tested the two approaches for handling the missing data problem while learning arm gestures by demonstration in humanoid robots. It is found that interpolating the signals first, using a shape invariant interpolant like the piecewise hermite interpolant gives better results. Next, we considered the learning from demonstrations problem in the task space. We saw how the GMM/GMR framework could be used to transfer a table-grasping skill from

the human to the robot.

We saw how human motion can be used for implicit human-robot communication. We considered humans and robots performing collaborative tasks, in which implicit communication is the most important medium of communication. First we considered a physical HRI task (collaborative table lifting task). Adjusting the leader/follower role for the robot autonomously was an important open problem. We contributed to a framework that utilizes human motion prediction to adjust the leader/follower role of the robot. The proposed framework consisted mainly of the reactive and proactive controllers. The proactive controller is based on a kalman filter for human motion prediction. Experimental results provided conclusive evidence that the proposed approach offered a definite improvement over simple reactive approaches. Further, we considered a higher level cooperative task in which human and the robot had to assemble various ingredients to prepare a dish in a cooperative cooking scenario. We presented a framework in which the robot could infer the humans intention by predicting his hand motion. Experimental analysis showed that inferring by predictions gave the robot lead time to start planning and acting.

6.1 Future Works

In this work, we have seen how human motion can be used for effective and natural human-robot interaction. For future works, it would be interesting to see how implicit and explicit communications can be used simultaneously for continuous and life-long human robot interaction. It would also be interesting to incorporate other media of communications such as speech, hand-gestures, facial expressions , etc. into a comprehensive framework. In this work, we mostly made use of the motion capture system for sensing human motion. It would be useful if the presented algorithms could be implemented to work for other sensing devices such as video cameras, Microsoft Kinect [47] or inertial motion sensors.

BIBLIOGRAPHY

[1] S. Thrun, "Toward a framework for human-robot interaction," *Hum.-Comput. Interact.*, vol. 19, pp. 9–24, June 2004.

[2] W. K. Mansour Rahimi, *Human-robot interaction*. Taylor and Francis, 1992.

[3] M. A. Goodrich and A. C. Schultz, "Human-robot interaction: a survey," *Found. Trends Hum.-Comput. Interact.*, vol. 1, pp. 203–275, January 2007.

[4] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration."

[5] Stefan and Schaal, "Is imitation learning the route to humanoid robots?," *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233 – 242, 1999.

[6] K. Dautenhahn and C. L. Nehaniv, eds., *Imitation in animals and artifacts*. Cambridge, MA, USA: MIT Press, 2002.

[7] A. Billard, Y. Epars, S. Calinon, S. Schaal, and G. Cheng, "Discovering optimal imitation strategies," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 69 – 77, 2004. ¡ce:title¿Robot Learning from Demonstration¡/ce:title¿.

[8] S. Calinon, F. Guenter, and A. Billard, "On learning, representing, and generalizing a task in a humanoid robot," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, pp. 286 –298, april 2007.

[9] K. Reed and M. Peshkin, "Physical collaboration of human-human and human-robot teams," *Haptics, IEEE Transactions on*, vol. 1, no. 2, pp. 108 –120, 2008.

[10] S. Calinon and A. Billard, "Incremental learning of gestures by imitation in a humanoid robot," in *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, HRI '07, (New York, NY, USA), pp. 255–262, ACM, 2007.

[11] H. Sakoe, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, pp. 43–49, 1978.

[12] N. Pollard, J. Hodgins, M. Riley, and C. Atkeson, "Adapting human motion for the control of a humanoid robot," in *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, vol. 2, pp. 1390 – 1397 vol.2, 2002.

[13] B. Dariush, M. Gienger, B. Jian, C. Goerick, and K. Fujimura, "Whole body humanoid control from human motion descriptors," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 2677 –2684, may 2008.

[14] U. Ales, A. Christopher, and R. Marcia, "Programming full-body movements for humanoid robots by observation."

[15] C. Ott, D. Lee, and Y. Nakamura, "Motion capture based human motion recognition and imitation by direct marker control," in *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on*, pp. 399 –405, dec. 2008.

[16] W. Suleiman, E. Yoshida, F. Kanehiro, J.-P. Laumond, and A. Monin, "On human motion imitation by humanoid robot," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 2697 –2704, may 2008.

[17] Vicon Motion Capture Systems, *http://www.vicon.com/*.

[18] Nao Humanoid Robot, Aldebaran Robotics, *http://www.aldebaran-robotics.com/*.

[19] H. Gray, *Gray's Anatomy*. Gramercy, 15th ed., 1988.

[20] F. N. Fritsch and R. E. Carlson, "Monotone piecewise cubic interpolation," *SIAM Journal on Numerical Analysis*, vol. 17, no. 2, pp. pp. 238–246, 1980.

[21] A. Thobbi and W. Sheng, "Imitation learning of hand gestures and its evaluation for humanoid robots," in *Information and Automation (ICIA), 2010 IEEE International Conference on*, pp. 60 – 65, june 2010.

[22] R. Dillmann, "Teaching and learning of robot tasks via observation of human performance," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 109 – 116, 2004. Robot Learning from Demonstration.

[23] S. Calinon, *Robot Programming by Demonstration: A Probabilistic Approach*. EPFL/CRC Press, 2009.

[24] S. Calinon and A. Billard, "A probabilistic programming by demonstration framework handling constraints in joint space and task space," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pp. 367 –372, 2008.

[25] D. Cohn, Z. Ghahramani, and M. Jordan, "Active learning with statistical models.," *Artificial Intelligence Research*, no. 4, pp. 129 – 148, 1996. Special issue on Information technology.

[26] G. Schwarz, "Estimating the dimension of a model.," *Annals of Statistics*, vol. 6, pp. 461 – 464, 1978. Special issue on Information technology.

[27] N. Hogan, "Impedance control: An approach to manipulation," in *American Control Conference, 1984*, pp. 304 –313, 1984.

[28] M. Rahman, R. Ikeura, and K. Mizutani, "Investigating the impedance characteristic of human arm for development of robots to co-operate with human operators," in *Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on*, 1999.

[29] Y. Maeda, T. Hara, and T. Arai, "Human-robot cooperative manipulation with motion estimation," in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, 2001.

[30] T. Flash and N. Hogan, "The coordination of arm movements: an experimentally confirmed mathematical model," *The Journal of neuroscience*, vol. 5, no. 7, pp. 1688–1703.

[31] B. Corteville, E. Aertbelien, H. Bruyninckx, J. De Schutter, and H. Van Brussel, "Human-inspired robot assistant for fast point-to-point movements," in *Robotics and Automation, 2007 IEEE International Conference on*, pp. 3639 –3644, 2007.

[32] P. Evrard and A. Kheddar, "Homotopy switching model for dyad haptic interaction in physical collaborative tasks," in *EuroHaptics conference, 2009 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics 2009. Third Joint*, pp. 45 –50, 2009.

[33] C. Smith and P. Jensfelt, "A predictor for operator input for time-delayed teleoperation," *Mechatronics*, vol. 20, no. 7, pp. 778 – 786, 2010. Special Issue on Design and Control Methodologies in Telerobotics.

[34] C. Passenberg, A. Peer, and M. Buss, "A survey of environment-, operator-, and task-adapted controllers for teleoperation systems," *Mechatronics*, vol. 20, no. 7, pp. 787 – 801, 2010. Special Issue on Design and Control Methodologies in Telerobotics.

[35] N. Jarrasse, J. Paik, V. Pasqui, and G. Morel, "How can human motion prediction increase transparency?," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 2134 –2139, May 2008.

[36] S. Clarke, G. Schillhuber, M. Zaeh, and H. Ulbrich, "Prediction-based methods for teleoperation across delayed networks," *Multimedia Systems*, vol. 13, pp. 253–261, 2008. 10.1007/s00530-007-0103-z.

[37] S. Miossec and A. Kheddar, "Human motion in cooperative tasks: Moving object case study," in *Robotics and Biomimetics, 2008. ROBIO 2008. IEEE International Conference on*, pp. 1509 –1514, 2009.

[38] E. Gribovskaya, A. Kheddar, and A. Billard, "Motion Learning and Adaptive Impedance for Robot Control during Physical Interaction with Humans," in *Proceedings of IEEE International Conference on Robotics and Automation*, 2011.

[39] D. Simon, D. L. Simon, and D. Viassolo, "Kalman filter constraint switching for turbofan engine health estimation. Discussion," *European journal of control*, vol. 12, no. 3, pp. 331–345.

[40] S. Gottschlich, C. Ramos, and D. Lyons, "Assembly and task planning: a taxonomy," *Robotics Automation Magazine, IEEE*, vol. 1, pp. 4 –12, sep 1994.

[41] T. Dean and M. Boddy, "An analysis of time-dependent planning," in *In Proceedings of the Seventh National Conference on Artificial Intelligence*, (Saint Paul, MN), pp. 49–54, 1988.

[42] E. Koechlin, G. Corrado, P. Pietrini, and J. Grafman, "Dissociating the role of the medial and lateral anterior prefrontal cortex in human planning," vol. 97, pp. 7651–7656, June 2000.

[43] E. Sisbot, L. Marin-Urias, R. Alami, and T. Simeon, "A human aware mobile robot motion planner," *Robotics, IEEE Transactions on*, vol. 23, pp. 874 –883, oct. 2007.

[44] D. Kulic and E. Croft, "Safe planning for human-robot interaction," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 2, pp. 1882 – 1887 Vol.2, 26-may 1, 2004.

[45] R. Alami, A. Clodic, V. Montreuil, E. A. Sisbot, and R. Chatila, "Task planning for human-robot interaction," in *Proceedings of the 2005 joint conference on Smart objects and ambient intelligence: innovative context-aware services: usages and technologies*, sOc-EUSAI '05, (New York, NY, USA), pp. 81–85, ACM, 2005.

[46] K. Sakita, K. Ogawara, S. Murakami, K. Kawamura, and K. Ikeuchi, "Flexible cooperation between human and robot by interpreting human intention from gaze information," in *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 1, pp. 846 – 851 vol.1, sept.-2 oct. 2004.

[47] Microsoft Kinect, *http://www.xbox.com/en-US/Kinect*.

[48] L. Lamport, *LaTeXUser's Guide and Reference Manual*. Addison Wesley, 2nd ed., 1994.

[49] H. Awano, T. Ogata, S. Nishide, T. Takahashi, K. Komatani, and H. Okuno, "Human-robot cooperation in arrangement of objects using confidence measure of neuro-dynamical system," in *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*, pp. 2533 –2538, 2010.

[50] S. Chernova and M. Veloso, "Confidence-based policy learning from demonstration using gaussian mixture models," in *Proceedings of the 6th international joint*

conference on Autonomous agents and multiagent systems, AAMAS '07, (New York, NY, USA), pp. 233:1–233:8, ACM, 2007.

[51] A. Thobbi and W. Sheng, "Imitation learning of arm gestures in presence of missing data for humanoid robots," in *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on*, pp. 92 –97, dec. 2010.

[52] Y. Gu, A. Thobbi, and W. Sheng, "Human-robot collaborative manipulation through imitation and reinforcement learning," in *Information and Automation (ICIA), 2011 IEEE International Conference on*, pp. 151 –156, june 2011.

[53] A. Thobbi, R. Kadam, and W. Sheng, "Achieving remote presence using a humanoid robot controlled by a non-invasive bci device," *ICGST International Journal on Artificial Intelligence and Machine Learning, AIML*, vol. 10, pp. 41–45, October 2010. Humanoid Robot, Tele-presence, Brain Computer Interface, Human Robot Interaction, Tele-operation.

[54] A. Thobbi, Y. Gu, and W. Sheng, "Using human motion estimation for cooperative manipulation," in *Intelligent Robots and Systems (IROS), 2011 IEEE International Conference on*, September 2011.

[55] H.-I. Lin and C. Lee, "Speed-accuracy optimization for skill learning," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pp. 2506 –2511, may 2009.

[56] I. Arel, D. Rose, and T. Karnowski, "Deep machine learning - a new frontier in artificial intelligence research [research frontier]," *Computational Intelligence Magazine, IEEE*, vol. 5, pp. 13 –18, nov. 2010.

[57] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, pp. 1345–1359, 2010.

[58] S. Kim, C. Kim, B. You, and S. Oh, "Stable whole-body motion generation for humanoid robots to imitate human motions," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pp. 2518 –2524, oct. 2009.

[59] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469 – 483, 2009.

[60] D. B. Grimes and R. P. Rao, "Creating brain-like intelligence," ch. Learning Actions through Imitation and Exploration: Towards Humanoid Robots That Learn from Humans, pp. 103–138, Berlin, Heidelberg: Springer-Verlag, 2009.

[61] J. J. Craig, *Introduction to Robotics: Mechanics and Control*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2nd ed., 1989.

[62] K. Hamahata, T. Taniguchi, K. Sakakibara, I. Nishikawa, K. Tabuchi, and T. Sawaragi, "Effective integration of imitation learning and reinforcement learning by generating internal reward," in *Intelligent Systems Design and Applications, 2008. ISDA '08. Eighth International Conference on*, vol. 3, pp. 121 –126, 2008.

[63] S. Calinon, F. D'halluin, E. Sauser, D. Caldwell, and A. Billard, "Learning and reproduction of gestures by imitation," *Robotics Automation Magazine, IEEE*, vol. 17, no. 2, pp. 44 –54, 2010.

[64] E. Gribovskaya, A. Kheddar, and A. Billard, "Motion Learning and Adaptive Impedance for Robot Control during Physical Interaction with Humans," in *Proceedings of IEEE International Conference on Robotics and Automation*, 2011.

[65] B.-N. Wang, Y. Gao, C. Zhao-Qian, and J.-Y. C. S.-F. Xie, "A two-layered multi-agent reinforcement learning model and algorithm," *Journal of Network and Computer Applications*, vol. 30, no. 4, pp. 1366 – 1376, 2007.

[66] A. M. Leslie Pack Kaelbling, Michael Littman, "Reinforcement learning: A survey.," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237 – 285, 1996. Special issue on Information technology.

[67] Y. E. Uri Kartoun, Helman Stern, "A human-robot collaborative reinforcement learning algorithm," *Journal of Intelligent Robotic Systems*, vol. 60, pp. 217–239, 2010.

[68] R. S. Sutton and A. G. Barto, "Reinforcement learning i: Introduction," 1998.

[69] M. B. Andrea Bauer, Dirk Wollherr, "Human-robot collaboration: A survey," *International Journal of Humanoid Robotics (IJHR)*, vol. 5, pp. 47 – 66, 2008.

[70] D. C. Bentivegna, C. G. Atkeson, and G. Cheng, "Learning tasks from observation and practice," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 163 – 169, 2004.

[71] S. Vijayakumar, T. Shibata, and S. Schaal, "Reinforcement learning for humanoid robotics," in *Autonomous Robot*, p. 2002, 2003.

VITA

Anand Rajiv Thobbi

Candidate for the Degree of

Master of Science

Thesis: THE ROLE OF HUMAN MOTION IN HUMAN-ROBOT INTERACTION

Major Field: Electrical and Computer Engineering

Biographical:

Personal Data: Born in Bangalore, Karnataka, India on March 14, 1988.

Education:

Received the B.S. degree from University of Pune, Pune, Maharashtra, India, 2009, in Electronics and Telecommunications
Completed the requirements for the degree of Master of Science with a major in Electrical and Computer Engineering, Oklahoma State University in December, 2011.

Name: Anand Rajiv Thobbi Date of Degree: December, 2011

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: THE ROLE OF HUMAN MOTION IN HUMAN-ROBOT INTERACTION

Pages in Study: 87 Candidate for the Degree of Master of Science

Major Field: Electrical Engineering

The aim of this work is to investigate the different ways in which humans can use their bodily movements for communicating and interacting with robots. The focus of this work is on observing, characterizing and predicting the human's arm and hand motion for human-robot communication(HRC). Such communication can be broadly classified as being explicit or implicit.

In this work, the robot learning from demonstrations (LfD) problem is studied as an example of explicit human-robot communications. We consider 2 cases such as -
(1) Robot learning to perform simple arm gestures : A framework based on Dynamic Time Warping (DTW) is proposed and implemented. Furthermore, we consider the case where the demonstrations contain missing data segments.
(2) Robot learning to perform tasks : The commonly used GMM/GMR framework is studied and implemented.

Implicit human-robot communication is prevalent when humans and robots work collaboratively. We consider two examples of collaborative tasks such as a -
(1) Joint table-lifting task : For this task to be truly successful, the robot needs to be able to decide its leader/follower role autonomously and dynamically. A framework in which the robot utilizes subtle cues from human motion to achieve this is proposed and evaluated. For enabling proactive leader-like behavior for the robot, it is necessary to learn the human motion model and generate next-state actions based on predictions generating from it.
(2) Co-operative cooking task: For proactively assisting the human, the robot should be able to infer his intent. We propose an approach in which the inference is based on long term predictions of the human's hand motion. The robot's actions are determined by the inferred human intent.

The theoretical ideas presented in this work are validated on an experimental platform consisting mainly of the Nao humanoid robot and the Vicon motion capture system. Qualitative and quantitative results demonstrate the efficacy of using human motion for both implicit and explicit human-robot communication.

ADVISOR'S APPROVAL (Dr. Weihua Sheng): _____